**Universidad Rey Juan Carlos**

Escuela Superior de Ciencias Experimentales y Tecnología

Departamento de Informática, Estadística y Telemática

# Sobre software libre

## Compilación de ensayos sobre software libre

### Grupo de Sistemas y Comunicaciones

**Editores**:

Vicente Matellán Olivera
Jesús M. González Barahona
Pedro de las Heras Quirós
Gregorio Robles Martínez

# Resumen "Sobre software libre"

"Sobre software libre" reúne casi una treintena de ensayos sobre temas de candente actualidad relacionados con el software libre (del cual Linux es su exponente más conocido). Los ensayos que el lector encontrará están divididos en bloques temáticos que van desde la propiedad intelectual o las cuestiones económicas y sociales de este modelo hasta su uso en la educación y las administraciones públicas, pasando por alguno que repasa la historia del software libre en los últimos años y los problemas que tiene que enfocar en un futuro próximo como es el caso de las patentes de programación. La obra incluye además una serie de artículos calificados como seminales dentro del mundo del software libre y firmados por Richard Stallman, uno de los grandes gurús del movimiento del software libre.

Éste es un libro pensado para un público objetivo muy amplio, no específicamente informáticos, sino también abogados, economistas, universitarios en general. Esto es así, porque la naturaleza de los contenidos, aunque técnica, está principalmente orientada en dar a conocer la vertiente más filosófica, sociológica y pragmática del software libre. En la actualidad no existe ninguna edición en nuestro idioma que permita a las personas interesadas (sobre todo las que no tienen una formación informática más allá de nivel de usuario) introducirse en la naturaleza y la filosofía del software libre, aún cuando esta temática está ganando en peso dentro y fuera del campo de la informática.

Esta obra ha sido concebida en su mayor parte por un activo grupo de profesores de la Universidad Rey Juan Carlos de Madrid, uno de los grupos pioneros en la adopción y difusión del software libre en España.

Se puede adquirir una copia del libro impreso en papel en el Servicio de Publicaciones de la Universidad Rey Juan Carlos o a través de la editorial Dykinson S.L. El ISBN de esta obra es el siguiente: 84-9772-402-X.

# Prólogo

Es común que los prólogos corran a cargo de un primera pluma en la materia que viene a contar lo mucho que le ha gustado el libro que han escrito unos segundones. Para desgracia del que esto escribe, pero probablemente para mayor satisfacción del lector, lo cierto es que en este libro se da la circunstancia contraria. Resulta difícil, si no imposible, encontrar un elenco de autores cuya influencia dentro del panorama del software libre en España (y en el mundo de habla hispana) sea mayor que la de los que firman la mayoría de los artículos incluidos en esta obra.

La colección se edita gracias a la convocatoria del Servicio de Publicaciones de la Universidad Rey Juan Carlos. No se trata de una casualidad, pues es una de las instituciones más activas en España en este tema, entre otras por las actividades del Grupo de Sistemas y Comunicaciones del que forman parte muchos de los autores.

Pero hablemos un poco del libro. *Sobre software libre* reune en un solo volumen sus ensayos más importantes publicados en diversos medios. Muchos de ellos van más allá de los aspectos técnicos o ingenieriles en los que en un principio incluiríamos al software libre, hasta el punto de que incluso me atrevería a decir que éstos tienen un papel secundario. Son más bien temas como la propiedad intelectual, las patentes de programación, el conocimiento libre o aspectos meramente económicos y sociales los que serán abordados desde diferentes perspectivas, pero siempre con la finalidad de mostrar lo que la *filosofía del software libre* puede aportar.

En clara concordancia con esta filosofía, la totalidad de los artículos de esta colección se publican bajo unas condiciones que permiten (foto)copiarlos y redistribuirlos. Y no sólo eso, los autores te invitan expresamente a que lo hagas, de forma que su obra llegue a cuantos más lectores mejor. ¡Por favor, cópianos!, claman todos los ensayos. ¡Por favor, repártelos!, piden los autores. Si entiendes esta filosofía, este libro te hará profundizar en ella. Si no, es una buena oportunidad para descubrir por qué creemos que estas ideas son beneficiosas para todos, incluyendo a los propios autores.

No puedo dejar de contar en este prólogo la razón por la que se ha elegido *Sobre software libre* como título de esta obra. Se trata de un guiño a uno de los primeros grupos españoles dedicados a la promoción y la discusión de temas relacionados con el software libre a principios de la década de los noventa y entre cuyos miembros encontraremos a algunos de los autores de este libro. Este grupo

se dio en llamar SoBre (resultado de cruzar SOftware y liBRE) y haber escogido como título *Sobre software libre* se ha de entender ciertamente como una especie de homenaje.

Finalmente, sólo me queda agradecer a los autores el haberme hecho descubrir tantas ideas apasionantes. Estoy seguro de que el lector, según vaya *devorando* las páginas de esta obra como he hecho yo, entenderá a qué me refiero. Feliz lectura, por tanto.

Junio de 2004 Gregorio Robles

# Índice general

## La Administración Pública



## La historia cercana



## Las piedras en el camino

## Artículos Seminales



## Apéndice

# Introducción

El software libre nació de la mano del propio software en la década de los años 60. Entonces las gigantescas máquinas a las que llamaban computadoras hacían uso de programas cuyo código fuente estaba a la vista de todos (los que querían verlo, por supuesto) y se podía distribuir libremente. Esto provocó que ya en esos tiempos, *prehistóricos* desde el punto de vista de la informática, existiera una pequeña comunidad de científicos y programadores que intercambiara código, a la vez que informes de errores e ideas. El software por entonces no era más que un valor añadido a las carísimas computadoras y se solía distribuir gratuitamente por los fabricantes.

La situación cambió radicalmente con el descenso del precio de las máquinas y sus componentes (el hardware) y la progresiva necesidad de un software más potente y con mayores funcionalidades. La ventaja competitiva que el intangible daba a las máquinas llegó hasta el punto en el que incluso había gente que estaba dispuesta a pagar dinero por él. Esto que en sí no es necesariamente malo, provocó sin embargo un giro radical en la industria informática: las primeras compañías exclusivamente dedicadas a la creación de software aparecieron en el horizonte y se hicieron fuertes en el mercado. En aras de maximizar beneficios (económicos y estratégicos), una de sus tácticas habituales era limitar hasta más no poder lo que el usuario podía hacer con el software que creaban.

De repente, algo tan natural hasta pocas fechas antes como compartir un programa o su código se convirtió en una práctica deleznable y que atentaba no sólo contra el creador del software, sino contra toda la industria del software y, por si acaso, también contra la sociedad y su bienestar. El lector, seguro que muy atento a los temas de actualidad, sabrá que este argumento se sigue utilizando de manera habitual una y otra vez en nuestros días por asociaciones de editores y grandes compañías de software: el que copia es nada menos que un *pirata*.

No fue hasta mediados los años 80, cuando Richard Stallman formalizó las ideas básicas del *movimiento* del software libre que está revolucionando la industria del software (y como se verá en este libro, puede que algo más). El software libre, tal y como lo conocemos hoy, dio sus primeros pasos con un manifiesto en favor de la libertad de expresión y un proyecto conocido hoy mundialmente, el proyecto GNU. Y con él, vio la luz probablemente una nueva forma de ver y entender el software y los bienes intangibles que se ha visto acelerada con la masiva implantación de Internet en las postrimerías del siglo XX y principios del actual.

Ha sido el binomio Internet-software libre (junto con otros ingredientes secundarios) el que ha propiciado uno de los cambios más radicales de las últimas décadas. Nótese que es difícil imaginarse el éxito del uno sin el otro. La mayor parte de la infraestructura de Internet se sustenta sobre código libre, mientras que las posibilidades colaborativas que ofrece Internet han sido vitales para el pleno desarrollo del software libre como elemento tecnológico y filosófico. Sin embargo, mientras el cambio tecnológico basado en Internet ha tenido una fuerte implantación en el mundo occidental, la mentalidad ligada al software libre está tardando algo más en calar en la sociedad. Pero no cabe duda de que paulatinamente va ganando en importancia.

Y es precisamente en este punto donde nos encontramos; en un mundo que está empezando a asimilar estos cambios y lo que conllevan. Los ensayos en este libro presentan y toman posición precisamente en algunos de los debates de más radiante actualidad que tienen que ver con estos aspectos. El lector podrá comprobar que el futuro tiene una clave en software libre y que ésta puede ser la llave hacia la sociedad post-moderna.

## Organización de la obra

La casi treintena de ensayos incluidos en este libro han sido agrupados en varios capítulos, de manera que aquéllos con una temática similar estén juntos. Esta clasificación ha sido realizada siguiendo criterios más bien subjetivos y es probable que el lector, tras una atenta lectura de todos ellos, realizara una ordenación diferente. No es infrecuente el caso de artículos en los que se tratan varios temas de manera simultánea. Cada capítulo cuenta con una fugaz introducción a la temática de la que trata, y en la que se resumen brevemente los aspectos más importantes de cada uno de los textos contenidos en el mismo.

El primer capítulo gira en torno a la propiedad intelectual, de la cual los derechos de autor y el copyright son parte. Es aquí donde se ahondará en aspectos del *¡cópiame, por favor!* y de las posibilidades que esta forma de concebir la propiedad intelectual ofrece.

En el siguiente se verán temas económicos, tanto de microeconomía como de macroeconomía, relacionados con el software libre. En el lado micro, se planteará la eterna pregunta de cómo se ganarán el pan los creadores de software si el bien que producen se puede redistribuir sin limitaciones, mientras que en la vertiente macro se analizarán los monopolios y sus consecuencias en el mundo del software.

En *El lado más técnico* están agrupados una serie de ensayos de carácter menos filosófico y más práctico. Allí se verán cuestiones más ligadas al desarrollador de software libre y al usuario *linuxero* medio. Cabe comentar que aún así, es una lectura poco tecnificada y apta para todos los públicos.

El capítulo dedicado a la educación y el conocimiento libre debatirá los conocimientos informáticos que los profesionales del mañana -hoy todavía en el

instituto o en la universidad- deberían adquirir. También habrá sitio para las iniciativas de difusión del conocimiento que algunas universidades están adoptando en tiempos recientes y que, a buen seguro, revolucionarán la enseñanza en un futuro no muy lejano (si no lo están haciendo ya). Para finalizar, con mayor grado de concreción eso sí, se discute acerca de los planes de estudio de las carreras de ingeniería informática.

La educación dará paso a una serie de artículos dedicados a la administración pública, en los que se presentan las inquietudes y deficiencias del sector público en cuanto a la adopción del software libre se refiere. Asimismo, también tienen cabida ideas relacionadas con las administraciones como pueden ser el acceso a la sociedad de la información, la difusión de conocimiento e incluso la tecnologización de la democracia.

Posteriormente nos detendremos en la evolución histórica del software libre en los últimos cinco años. El repaso incluye novedades, carencias y posibles peligros y permite observar cómo el software libre ha ido creciendo y conquistando terrenos hasta hace poco impensables.

Aún así, en el horizonte se vislumbran en un futuro no muy lejano algunas fuentes de serios problemas para el software libre. Dada su importancia, se les ha dedicado íntegramente un capítulo titulado *Las piedras en el camino*. Curiosamente las cuestiones que trataremos y que más afectan al software libre no son de naturaleza técnica, como cabría esperar. En estos artículos se hablará de patentes, de leyes de propiedad intelectual más restrictivas, de legislación para Internet, etc.

En una obra con ensayos sobre software libre no pueden faltar algunos documentos seminales, auténticas piedras filosofales de este movimiento. En ellas, el lector podrá encontrar las raíces más profundas de este movimiento. Así, es probable que el lector novato en cuestiones de software libre quiera empezar por ellos para introducirse en lo que es el software libre y en los pilares básicos de la filosofía que hay detrás. En todo caso, estos artículos suponen un broche de oro a esta colección.

Finalmente, se ha incluido en un apéndice un glosario de términos y acrónimos con el claro objetivo de facilitar la comprensión de los artículos incluidos en esta obra al mayor número de personas, independientemente de su grado de formación informática y de su conocimiento del mundo del software libre. Los términos se pueden distinguir fácilmente, ya que se ha utilizado un tipo de fuente diferente al normal, como ya se ha venido haciendo en esta introducción.

# La propiedad intelectual

Ya desde pequeños nos familiarizamos rápidamente con el concepto de propiedad. Sin ir más lejos, yo me acuerdo cómo entró en mi vida: en los recreos del *cole*. A esa edad todavía tenía la sana costumbre de coleccionar cromos, que se intercambiaban al melódico *si le, no le*[1] entre clase y clase. Un cromo o era tuyo o era del amiguete con el que los cambiabas, pero no podía ser de los dos a la vez; al fin y al cabo, ese cromo sólo podía acabar en la colección de uno de los dos. Como muy tarde, era justamente a la hora de pegarlos en el álbum cuando te dabas cuenta de que el cromo era propiedad tuya y de nadie más.

Si me hubieran hablado entonces de la propiedad de bienes que no se pueden tocar, no le hubiera encontrado ningún sentido. Probablemente, hoy todavía no se lo encuentre. Y es que desde hace más o menos un par de siglos, el concepto de propiedad que tenemos para los bienes tangibles se ha extendido a los intangibles, lo que ha dado paso a la propiedad intelectual. Cuando se planteó esta analogía, los bienes intangibles se podían copiar, pero esto era una práctica cara (y, a veces difícil), al alcance de unos pocos. Por eso, la idea de propiedad que dábamos a un bien intangible (pongamos por caso un libro o un CD) se asemejaba mucho a uno tangible: no lo podías compartir sin quedarte sin él. Aunque ciertamente, eso fue hace mucho tiempo. Hoy, con los bajos costes de copia, literalmente cualquiera puede conseguir una copia fidedigna del original. Si los cromos fueran un bien intangible, el *si le, no le* de los recreos estaría en vías de extinción, ya que obtener una copia idéntica del cromo de mi amigo no sería nada difícil y el coste sería despreciable.

Precisamente uno de los aspectos más llamativos del software libre es su desafío a la concepción imperante sobre el derecho a la propiedad intelectual, en especial del *copyright*. Resulta paradójico para muchos cómo el software libre se vale precisamente del *copyright* -las licencias que rigen las condiciones de uso y (re)distribución del software- para conseguir justamente lo contrario que lo que ha venido a ser común en los últimos siglos. De eso y de mucho más es de lo que se va a hablar en este capítulo.

El primer artículo, *Copiar o no copiar, ¿he ahí el dilema?*, analiza este aspecto utilizando para ello la analogía de la máquina de duplicar pan. Es interesante observar cómo el modelo que propone el software libre resulta beneficioso para todos, incluso -para incredulidad de muchos- para el propio autor. Y resulta

[1] Ciertamente esta fórmula contiene un leísmo propio de la zona donde crecí.

que en contraposición con el modelo de *copyright*, el nuevo modelo (denominado popularmente como copyleft en un hábil juego de palabras tan dado a los anglosajones) funciona mejor si copias el software.

Ya he comentado en el prólogo que este mismo libro sigue la filosofía del copyleft. Y es que el modelo adoptado por el software libre se puede trasladar con algunos matices a otros bienes intangibles. En *Músicos, compositores y rentistas* se aplicará sobre la música, indagando en los diferentes roles que existen en la creación de obras musicales y las implicaciones que un modelo similar al copyleft tendría en la industria musical. *Software, mentiras y cintas de vídeo* lo hará seguidamente con el cine, haciendo cábalas de cómo sería el cine libre (modificable y redistribuible).

El último ensayo de este tema, *Y la información será libre... ¿o no?*, muestra dos posibles escenarios en el futuro: uno en el caso de que las leyes de propiedad intelectual hubieran desaparecido y otro en las que se hubieran añadido más restricciones a las mismas. En ambos casos se trata simplemente de una extrapolación de movimientos que podemos observar en la realidad cotidiana de nuestros días.

# Copiar o no copiar, ¿he ahí el dilema?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

Uno de los aspectos del software libre que más sorprenden al recién llegado es que el autor no sólo permita que su trabajo sea copiado y redistribuido libremente, sino que además anima al usuario a que lo haga. En general, tanto el autor como el usuario como quien recibe el programa que se copia quedan contentos. Y sin embargo, esto parece ir en contra de una idea que hemos oído muchísimo en los último años: *no se debe copiar software*. ¿Qué está ocurriendo aquí? ¿Habías pensado alguna vez sobre ello?

## La máquina de duplicar pan

Olvidémonos por un momento de GNU/Linux, del software (libre o no) y de la copia de programas. Y fabulemos un poco. Imaginemos que, en alguna parte del mundo, en una prestigiosa universidad, alguien construye un invento completamente imprevisto: la máquina de duplicar pan. Es una máquina maravillosa. Casi no consume energía, la puede manejar cualquiera, se puede construir por millones a bajo coste, y no necesita materia prima ni mantenimiento. Su funcionamiento es simple: introduces una pieza de pan por un lado, y salen dos por el otro. La segunda es indistinguible de la primera: igual de sana, igual de nutritiva, con el mismo sabor. Y el proceso se puede repetir indefinidamente, usando las piezas de pan originales o las nuevas que ha producido la propia máquina. Muchos ya ven los problemas mundiales de hambre resueltos por fin. Se piensa inmediatamente en los duplicadores de lechugas, filetes, zanahorias, lenguados y otros muchos alimentos....

Pero antes de que nada de esto ocurra, comienza una campaña de publicidad en todos los medios. En ella aparecen drogadictos, asesinos, ladrones... y duplicadores de pan. El lema de la campaña es: *Cada vez que alguien utiliza el duplicador de pan, todos perdemos*. La campaña está dirigida por las asociaciones de panaderos, con dinero de toda la industria de la alimentación. Aparecen en todas las cadenas de televisión, en todos los periódicos, en todas las emisoras de radio opiniones a favor de los panaderos y en contra de los que están empezando ya a duplicar pan. Se plantean cuestiones como *Si se permite la máquina de duplicar pan, ¿quién asegurará que tendremos innovación, y nuevos tipos de pan?* o *Si no se prohibe la máquina de pan, ¿de qué van a vivir los panaderos?* Inmediatamente se proponen legislaciones que prohiben la fabricación, comercialización y uso de máquinas duplicadoras de pan, y se empieza a considerar éticamente malo duplicar pan... Con el tiempo, las legislaciones se ponen en vigor, se crean ramas específicas de la policía para perseguir la copia ilegal de pan, y los panaderos, ya

organizados, empiezan una campaña para que los productores independientes de pan tengan que pagar derechos por las recetas de los tipos de pan más habituales.

## ¿Una fábula sin sentido?

Bueno, volvamos a la realidad. ¿Por qué cuento todo esto? ¿Es que tengo una vena oculta de cuentacuentos con extrañas moralejas? No creo... Lo de las barras de pan se lo oí hace tiempo a Richard Stallman, y resulta que es algo que tenemos aquí ya... casi. Si en lugar de pan hablamos de programas, la máquina duplicadora la tenemos casi todos en nuestro PC. De hecho, tenemos varios tipos de ellas: Internet, disquetera, duplicador de CDs. Lo cierto es que desde hace unos años, los humanos disponemos de algo único en la historia: tenemos máquinas que pueden duplicar no sólo programas sino cualquier tipo de información a un coste prácticamente ridículo.

Hasta hace poco (en términos históricos) copiar información era caro y difícil. Los copistas medievales dedicaban su vida a ello, y sólo copiaban unos cuantos pergaminos a lo largo de ella. La imprenta mejoró mucho las cosas, pero no todo el mundo tenía una imprenta, y con ella no era económico hacer pocas copias. Y para distribuirlas y elegir qué se copiaba nació una inmensa industria: la industria editorial. La imprenta y esta industria, junto con otros factores, ayudaron a que la producción de información escrita creciera como nunca.

Cuando aparecieron los ordenadores y se empezaron a distribuir programas, se utilizó una organización similar a la del material impreso. Y así nacieron grandes empresas de software cuya labor es muy similar a la de los editores. Para que estas empresas pudieran funcionar, la sociedad, en todo el mundo, decidió que había que aplicar a los programas una legislación similar a la de los libros, y en general se prohibió la copia de programas si el autor no daba permiso. Y el autor, normalmente, no daba permiso. En algunos países esto pareció poco, y se decidió que también se podía prohibir a los propios autores aplicar ciertas ideas a la hora de hacer programas. Las leyes que prohiben la copia de programas son las de derechos de autor (*copyright*) y las que prohiben la utilización de ciertas ideas son las de propiedad industrial (patentes). La motivación para estas leyes, en el caso del software, es similar: se supone que favorecen la innovación, aseguran que todos tengamos suficiente cantidad y calidad de programas, y permiten que los programadores vivan dignamente.

Pero todo esto no invalida el hecho fundamental: tenemos una máquina que permite duplicar los programas a coste prácticamente cero. Si la sociedad decide no usarla tiene que ser por poderosas razones. Si en algún momento quedase claro que puede producirse suficiente cantidad y calidad de programas sin prohibir la copia... ¿qué motivo tenemos para prohibirnos ese derecho?

## Y, sin embargo, se mueve

Hace ya muchos años, en este océano de presión contra la copia de software, hubo un grupo de gente que nunca dejó de compartir sus programas, y de dejar que otros los repartieran a quien quisieran. Con el tiempo, esta comunidad creció y creció. De producir sólo unas cuantas herramientas para programadores pasó a generar miles y miles de programas para todos los públicos, desde procesadores de texto hasta navegadores de web. De estar compuesta fundamentalmente por voluntarios trabajando en su tiempo libre pasó a ser un hervidero de distintos tipos de gente, muchos pagados por empresas, muchos con sus propias empresas. De contar sólo con cantidades mínimas de dinero obtenidas vendiendo camisetas se pasó al dinero de las firmas de capital riesgo, de fondos de pensiones y de accionistas. De los individuos independientes y las empresas unipersonales se pasó a tener también multinacionales en el juego. Y de ser un puñado de desarrolladores en algunos lugares concretos se pasó a una comunidad de miles y miles de desarrolladores repartidos por todo el mundo.

Y a pesar de este proceso, que ha cambiado tantas cosas, que ha causado tantas tensiones, y que ha producido tantos programas, algo quedó siempre claro: si tú recibes un programa libre, tienes derecho a copiarlo para tus amigos, para tus clientes, para quien sea. Y eso es bueno para ti, para tus amigos, para tus clientes... y para el autor del programa.

Es fácil entender cómo este mecanismo te beneficia a ti, a tus amigos y a tus clientes. Es más largo de entender cómo beneficia al autor del programa, pero es un hecho que es así. Muchos programadores viven ya del software libre, bien recibiendo ingresos directamente de quien lo usa, o bien cobrando un sueldo en alguna empresa que genera sus ingresos con un modelo de negocio basado en el software libre. Explicar cómo puede suceder esto, de dónde sale la financiación y cómo se puede crecer y ganar dinero si no se cobra por copia vendida es largo de explicar, y probablemente necesitaría otro articulo entero sólo para empezar[1]. Pero a estas alturas es un hecho que esto ocurre, luego la pregunta ya no es tanto *¿Es posible?* sino *¿Cómo es posible?*.

Pero aún hay más. El modelo del software libre no sólo permite que tú copies y redistribuyas los programas que recibas. El modelo del software libre funciona mejor si lo haces. Cada vez que estás copiando un CD de GNU/Linux para un amigo, estás ayudando a que el software libre funcione mejor. Cada vez que un grupo de usuarios hace una tirada de CDs de Debian y los vende a bajo precio en una fiesta de instalación, está ayudando a que funcione el modelo del software libre. Cada vez que Red Hat, Mandrake o SuSE venden un CD en un hipermercado, están ayudando a que todos tengamos más y mejor software libre. De nuevo explicar esto es complicado, pero aquí sí es fácil sugerir ideas. Mayor número de usuarios supone un mayor mercado. Supone acercarse más a ser el

[1] Nota del editor: Más adelante, en esta misma colección, hay unos cuantos artículos dedicados a cuestiones económicas que abordarán esta problemática.

número uno en ese nicho. Supone ser el estándar de referencia. Supone mucha gente interesada en aprender a usar ese programa, y mucha gente y muchas empresas dispuestas a pagar por servicios alrededor de ese programa. Muchos desarrolladores interesados en colaborar con mejoras y corrección de fallos. Cada vez que das una copia de un programa libre a un amigo, estás ayudando a que toda esta enorme rueda gire... en la dirección que más te beneficia.

## La gran pregunta

Naturalmente, si todo esto es cierto (y hay millones de usuarios que dicen que sí es cierto), tenemos un modelo de producción de programas que ha demostrado que es capaz de producir suficiente cantidad y calidad para mucha gente. ¿Será capaz de generar suficiente calidad y cantidad para la mayoría de la gente? ¿Para toda la gente? Sólo el tiempo lo dirá, claro. Quizás todo esto no sea más que una burbuja que se desinfle en unos meses, y de la cual nadie se acuerde dentro de unos años. Quizás ninguna empresa sea capaz de encontrar un modelo de negocio que le permita tener ingresos saneados de forma estable. Quizás deje de innovarse en el software libre, y quizás nunca haya programas libres en muchos nichos. Pero si la tendencia actual continúa, la situación será más bien la contraria. Si seguimos por el camino de los últimos años, dentro de no mucho tiempo tendremos una saneada industria del software libre, con una poderosa comunidad de desarrolladores y usuarios satisfechos alrededor.

Y si todo es cierto, podemos volver a la gran pregunta, y decir: ¿qué motivo tenemos para renunciar al derecho a copiar programas? ¿Realmente es preciso prohibir la copia para que tengamos el software que necesitamos? Y más allá: si podemos tener el software que necesitamos sin prohibir la copia (ni de programas ni de ideas), ¿no sería mejor permitirla siempre, puesto que en ausencia de otros problemas los usuarios ganarían mucho?

## ¿Un cambio de tendencia?

Aún estamos dentro de una tendencia que parece llevarnos hacia más y más restricciones legales a nuestro derecho a copiar software. Las legislaciones sobre derechos de autor en informática son cada vez más estrictas, y las penas que se aplican son cada vez más grandes. Y quizás esto sea bueno para el desarrollo del software libre: cuanto más prisionero se encuentre un usuario de las empresas del software propietario, más motivado estará para probar con las opciones libres.

Pero en este entorno, es importante no perder de vista la situación de base: el único motivo para perseguir la copia es que eso sirva para motivar a los autores a desarrollar más y mejores programas. La única razón por la que en las sociedades democráticas podemos permitir que se nos obligue a pagar a un particular por algo que podríamos hacer gratis es porque eso beneficia a la sociedad en su conjunto (en el caso del software, generando suficientes recursos para garantizar

que se desarrolle más software de calidad). Si en algún momento esto dejarse de ser cierto, no habría muchos motivos para esta prohibición, ¿no crees?

Y al menos hay una comunidad (la del software libre) en la que esto ha dejado de ser cierto. Por ahora, aún no se ha demostrado el caso general, pero ya tenemos casos particulares. Así que atención a los próximos años... y ojo a las ideas preconcebidas. Si tienes un derecho, no renuncies a él sin buenos motivos. Sigue usando software libre y da una oportunidad a la realidad para cambiar... hacia mejor.

# Músicos, compositores y rentistas

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 10, pág. 12-13, Agosto de 2001

La demanda contra Napster de las casas discográficas, realizada por la RIAA (*Recording Industry Association of America*), es la punta del iceberg en el cambio que se está produciendo en el mundo de la música como consecuencia de las nuevas tecnologías. ¿Cuál es ese cambio y por qué persigue la RIAA con tanto empeño el cierre de Napster?

Haciendo un recorrido por la página web de la RIAA se pueden leer multitud de alegatos en favor de la creatividad, de su misión en defensa de los artistas, etc. ¿Es realmente eso lo que defiende la RIAA o el equivalente español, la SGAE?

Realmente ambas asociaciones mezclan, desde mi punto de vista de manera interesada, diversos perfiles. Tal es el caso, por ejemplo, con los perfiles de autor y editor. Sus derechos y sus intereses son distintos. Aunque esas distinciones son aplicables también a la literatura o el cine, me voy a centrar en el caso de la música. Simplificando veo tres roles básicos: músicos (interpretes de música), compositores (creadores de música) y distribuidores.

## ¿Quiénes son los músicos?

¿Cómo se gana la vida un músico? Pues, en mi opinión, tocando música. ¿Quién paga o debería pagar a los músicos? Los que van a escuchar su música. Si van miles de personas (por ejemplo en un gran auditorio, estadio, etc), o cobran entradas muy caras (por ejemplo Barbara Streisand en Las Vegas) pueden ganar mucho dinero. Creo que ésta es la profesión más extendida en el mundo de la música, por ejemplo en España estoy seguro que existen varios miles de personas que se ganan la vida como músicos: los que tocan en las fiestas de los pueblos, la banda que toca en las bodas, los que dan ese fondo musical al café del centro, etc. etc.

¿Cómo se gana la vida un compositor? Pues obviamente componiendo música. ¿Quién paga a los compositores? Pues tradicionalmente los músicos, sólo hay que recordar los tiempos en los que al presentar un artista se especificaba quién había compuesto la música.

Hoy en día mucha más gente paga por componer música: las televisiones por las cortinillas, los publicistas por los anuncios, las productoras de televisión por las sintonías de sus series, etc. De nuevo si son muy buenos, cobran más por sus composiciones.

El caso más general en los grupos de música moderna (las operas y sinfonías siguen siendo mayoritariamente de encargo) es la mezcla de músicos y compositores. Así, los grupos de pop, rock, etc. suelen ser autores de sus canciones,

pagando por los arreglos si los necesitan. Por cierto, esas figuras, la del arreglista, el productor, etc. que cobran por sus servicios, y desde luego son también parte del proceso creativo, parecen no tener importancia.

Por último, ¿Cómo se gana la vida un distribuidor de música? Pues *enlatando* la música y vendiendo esas latas con un margen de beneficio. Hace algunos años en forma de discos de vinilo, después en forma de cintas y, desde los años 90, en CDs.

## ¿Cuál es el problema ahora?

¿Cuál es el problema entonces? Pues que a estos últimos se les está encogiendo el mercado. Los músicos siguen cantando, los compositores siguen componiendo, etc. Son los vendedores de CDs a 3.000 pts. los que están preocupados porque ahora cualquiera puede hacer su trabajo por 100 pts. ¿Por qué cobran 3.000 pts. por un CD? Pues supuestamente por los *gastos de promoción y distribución* del disco. Sin embargo, hoy esos gastos deberían ser mucho menores, pues Internet puede hacer esas cosas muy bien.

Como su queja de que se les acaba el chiringuito no es defendible, tratan, y parece que lo consiguen, de escudarse en que la *copia* perjudica a la música en general y a los músicos en particular. Cuando en mi opinión a los que perjudica fundamentalmente es a ellos. ¿Cuántos músicos de esas decenas de miles que he citado antes se ganan la vida con los derechos que les paga la SGAE? Estoy por asegurar que no son más que unos pocos cientos los que pasan de los 3 millones de pts. anuales en España por derechos de autor, por ejemplo.

Como defender los derechos de esos pocos cientos de *rentistas* no tendría muy buena prensa, hay que crear el concepto social de *compartir la música es malo*. Así, la SGAE ha pagado unos anuncios terribles en la televisión española basados en el famoso *Sex, Drugs and Rock&Roll*. Sexo: una violación. Drogas: una sobredosis. Rock&Roll: CDs *copiados*. ¡Comparan una violación con copiar CDs!

Venimos de un mundo en el que la gran mayoría ve perfectamente normal prestarle a sus amigos un CD para que vean lo bueno que es (aunque eso es probablemente igual de ilegal que hacerlo a través de Napster). Vamos hacia uno en el que se accederá a la música de forma remota, desde nuestro propio almacén de música (disco duro o similar), desde un servidor central, o desde un tipo-Napster. Obviamente la gente seguirá viendo normal prestarle su música a otro (Napster), sobre todo si ese alguien también me presta la suya a mí. Ésa es la imagen que la RIAA, la SGAE, etc. quieren cambiar. Nos quieren llevar a un mundo como el que describía Richard Stallman en su *Derecho a leer*[1].

Veámoslo con otro ejemplo: los humoristas se ganan la vida contando chistes, actúan en televisión y en locales, etc. ¿A alguien se le ocurriría prohibirnos repetir

[1] Nota del editor: el artículo al que se hace referencia aquí, se puede leer también en este volumen, ya que ha sido incluido entre los artículos seminales.

un chiste que hemos oído? ¿Alguien se cree que por tararear una canción se van a componer menos o peores canciones? ¿y por tocarlas con tu guitarra en casa? ¿Y por tocarlas con un grupo de amigos en el garaje de tu casa? ¿Cómo tiene la SGAE el atrevimiento de pedir a los músicos que tocan en una boda un canon por las canciones que tocan? El trabajo de esos músicos es animar esa fiesta, con canciones, pero también con su presencia, con su *actuación*. ¿Por qué tienen que pagar por tocar *Los Pajaritos* y no por contar un chiste?

Existe otro argumento que personalmente siempre me ha resultado curioso: *No hay otra forma de hacerlo*. Es el viejo *el fin justifica los medios*. En este caso el argumento en favor de limitar el instinto humano de compartir información, es *que no se podría tal o cual cosa por falta de dinero*, ya sea un disco o una película. Cada vez que oigo ese argumento me imagino a la persona que lo dice, vestida de faraón egipcio, razonando con toda seriedad en favor de la esclavitud: *cómo vamos a construir pirámides sin esclavos*.

Puede ser cierto, quizás no se puedan construir pirámides sin esclavos, ni hacer tal grabación multimillonaria sin prohibir la copia. Pero no es motivo para cercenar un derecho, bien sea el de la libertad para elegir dónde trabajar, o para compartir información. Si la sociedad necesita pirámides, encontrará la forma de construirlas sin tener que esclavizar a la población, de hecho ahora se construyen edificios más grandes y sofisticados como son los rascacielos. Si necesita películas espectaculares encontrará la forma de hacerlas (por ejemplo con escenarios virtuales) sin tener que obtener el retorno de la inversión a base de cercenar el derecho a compartir información

## ¿Hay alternativas?

¿Cuál es la alternativa al modelo de la copia? Probablemente un sistema similar al del software libre es fácilmente implantable en el caso de la música. Eso sí, habría que comenzar una campaña de educación: una cosa es la *autoría* de una canción, otra la realización de copias de los soportes en los que se distribuye esa canción. Cada uno de los implicados en la generación de música cobraría, más o menos según su calidad (o suerte), según hemos visto antes: los autores por componer, los cantantes por cantar y los distribuidores por distribuir.

Un tercer asunto es la modificación de una canción (las versiones). Richard Stallman es el autor original del editor Emacs, y se le reconoce por ello. Cualquiera puede hacer copias de Emacs, distribuirlas, venderlas, etc. Incluso se han realizado versiones de Emacs, como XEmacs, que probablemente no son del agrado del autor original. ¿Puede un autor de software libre negarse a que alguien modifique su software? Pues no, es la forma que tenemos de tener un software mejor. ¿Por qué un autor de música se arroga el derecho a que una canción es algo *suyo*? Desde luego es libre de mantener *su* versión, exactamente igual que Richard Stallman puede mantener su versión de Emacs.

Este tercer asunto es desde mi punto de vista el más conflictivo, pues para muchos autores sus obras son algo intocable y no perfeccionable. No lo comparto, pero sí creo que sería mejor separarlo de los otros para resolver al menos algún problema.

## Algunos enlaces

Para terminar, me gustaría presentaros un par de enlaces relacionados.

Courtney Love, la actriz y cantante estadounidense, escribió un manifiesto en el que hablaba sobre el negocio de la música y los verdaderos piratas que viven a costa de los autores. Empieza así: "Hoy quiero hablar sobre pirateo y música. ¿Qué es el pirateo? El pirateo es el acto de apropiarse del trabajo de un artista sin pagarle por ello. No estoy hablando acerca del software tipo Napster. Estoy hablando sobre los contratos de las principales compañías discográficas."El artículo se publicó en Salon.com[2] y los comentarios en BarraPunto[3].

Richard Stallman pronunció una conferencia en el MIT, titulada *Copyright and Globalization in the Age of Computer Networks*. Stallman proponía que se pagasen precios diferentes para adquirir beneficios diferentes: sólo se debe pagar un precio alto (cesión del derecho a copiar) cuando el beneficio que vaya a obtener la sociedad sea alto. Así ocurrió al aparecer con la imprenta: la sociedad cedió su derecho a copiar, entre otras cosas, porque no podía ejercerlo fácilmente (la imprenta costaba mucho). Hoy en día, sin embargo, hacer copias es fácil, por lo que debemos renegociar el precio que pagamos. Puedes leer conferencia original[4] y los comentarios en BarraPunto[5] al respecto, si así lo deseas.



[2] http://www.salon.com/tech/feature/2000/06/14/love/print.html

[3] http://barrapunto.com/articles/100/06/16/0817239.shtml

[4] http://web.mit.edu/m-i-t/forums/copyright/

[5] http://barrapunto.com/article.pl?sid=01/05/07/1015242

# Software, mentiras y cintas de vídeo

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 19, pág. 12-13, Abril de 2002

Ya está aquí. La amenaza a otro de los negocios más exportadores de los EEUU en todos los sentidos está en marcha. Para los que no lo tengan claro no hablo del acero, tan de moda estos días por los aranceles que ha impuesto el gobierno del país abanderado del libre comercio. Uno de los primeros negocios de los EEUU es la exportación de su cine. Las factorías de Hollywood obtienen miles de millones de euros por ventas de entradas en todo el mundo, pero sobre todo obtienen más en derechos de retransmisión en televisión, venta de cintas y DVDs, y *merchandising* en general.

Igual que Napster amenazó la industria de la música hace algún tiempo, el intercambio de películas en formatos como DivX tiene muy preocupados a los peces gordos de las multinacionales del cine. Tanto es así, que muchas han sido reacias a publicar sus títulos en DVD (a pesar de los precios que cobran).

De hecho, la historia de los propios DVD se ha visto terriblemente entorpecida por los miedos de las grandes productoras. En primer lugar por sus peleas sobre los formatos y su seguridad, de donde vienen males tan graves como las zonas geográficas de comercialización de los DVD que dividen artificialmente el mercado. Estas zonas están ahora denunciadas por la Unión Europea como prácticas contra la competencia[1]. Lo que lleva a uno a preguntarse cómo es que no se habían enterado antes (pero esto lo dejaremos para otro día)..

Otro problema que ha causado el miedo de los estudios es la persecución a desarrolladores de software libre que han realizado implementaciones de DeCSS para poder ver DVDs en Linux, etc.

En cualquier caso, esto no es más que una preocupación legítima con el estado actual de la legislación sobre propiedad intelectual, que por otra parte parece difícil de cambiar (aunque ya he argumentado en otros artículos que es indispensable). A mí me surgen ahora algunas preguntas: ¿Podría existir el cine libre? ¿Cómo sería la distribución de contenidos audiovisuales en el futuro? ¿Cómo se organizaría esta industria?

## Cine libre

Antes de nada, ¿tiene sentido que exista el cine libre? Entendiendo como *libre* que pueda ser libremente distribuible, modificable... es decir, cine con garantías similares a las del software libre. En primer lugar habría que ver si tiene sentido

[1] http://barrapunto.com/article.pl?sid=01/06/14/076243

que sea libre en el mismo sentido que el software. Por ejemplo, si tiene sentido que sea *modificable*.

Hay mucha gente que piensa que el arte no debe poder ser modificable, gente cuyas opiniones respeto mucho, como Richard Stallman por ejemplo. Yo, sin embargo, creo que el arte no es más que otra forma de producción intelectual y que, por tanto, está en la naturaleza humana el reaprovecharla, modificarla, usarla en definitiva de formas no previstas por sus autores. Los autores siempre podrán mantener *sus* versiones de las obras que han realizado, pero es difícil (imposible diría yo) impedir que sus obras se tomen como modelos, como inspiración o como contraejemplo, incluso que usen de formas contrarias a sus creencias u opiniones. Así, por ejemplo, hay versiones de las Meninas de Velázquez de renombrados pintores posteriores y con muy diversas interpretaciones, por no hablar de las versiones *profanas* de los personajes de Disney. En particular, en el caso del cine reaprovechar escenas ya grabadas, personajes virtuales o guiones me parece de nuevo difícil de evitar.

En segundo lugar habría que analizar si es posible que exista cine *redistribuible*. Muchos opinan que no (estos comentarios en BarraPunto son una muestra[2]). Sin embargo, yo estoy convencido de que sí. De entrada, existe un un tipo de cine al que le interesa mucho que se reproduzca cuantas más veces mejor sin cobrar: los anuncios. Las empresas anunciantes contratan a conocidos directores, guionistas y famosos actores, invierten en decorados y efectos especiales y luego además se gastan dinero en que su *obra* se reproduzca.

Además estoy convencido de que muchos actores, directores, y en general personas en todas las profesiones involucradas en la generación de contenidos cinematográficos están interesados en formar *redes colaborativas*, al estilo de las que existen alrededor de los proyectos de software libre, que les permitan ejercitar su creatividad con mayor libertad que las que les dan las productoras convencionales, por no hablar de aquéllos más interesados en aspectos sociales o políticos.

## El futuro del cine

¿Cuál sería la organización, el funcionamiento, de este cine libre? Voy a dar rienda suelta a mi vena utópica y voy a tratar de describir una posible situación. Para empezar, yo me imagino que en un futuro cercano los personajes virtuales serán grandes estrellas, fundamentalmente porque cobran menos y pueden hacer muchas películas a la vez, rentabilidad en definitiva.

Hoy ya tenemos personajes virtuales famosos, como Jar Jar Binks de la saga de *Star Wars*, que son más conocidos que la mayoría de los actores de carne y hueso. Hay incluso películas completas realizadas con actores virtuales como *Final Fantasy* que han sido grandes éxitos de taquilla. De hecho, existen personajes virtuales mucho más conocidos como los ya citados dibujos animados.

[2] http://barrapunto.com/article.pl?sid=02/02/24/2122222

Estos personajes virtuales presentan muchas ventajas con respecto a los de carne y hueso: disponibilidad, coste, versatilidad, etc. Sin embargo tienen los mismos problemas que el software propietario: básicamente tienen dueño. Esto quiere decir que su uso sólo puede realizarse mediante licencia. Hasta ahora este modelo ha funcionado bastante bien, por ejemplo todo el *merchandising* debe venderse bajo licencia, aparecen en anuncios, etc.

Este modelo tiene fundamentalmente el mismo problema que el software propietario o de la música. Los dueños de las licencias se quejan de la piratería. Los usuarios se quejan de los altos precios de venta teniendo en cuenta los costes de fabricación.

Si la comunidad empieza a desarrollar personajes virtuales libres estoy convencido que a medio plazo serán mejores en todas sus facetas: desde las puramente informáticas como su movimiento o el *ray-tracing*, hasta las más comerciales como su conocimiento público o la creación de su personalidad virtual.

Me imagino una situación futura en la que, usando los personajes virtuales, unos cuantos guionistas distribuidos por el mundo acuerden un guión y se pongan a trabajar en la película. Una vez realizada se podrían realizar incluso diferentes montajes, añadir escenas exitosas de otras películas, modificar planos de escenas ya realizadas, etc. En resumen, el mismo tipo de flexibilidad que hoy disfrutamos en el mundo del software libre.

Desde luego, la mayoría de estas cosas pueden realizarse también con material grabado con actores humanos y con equipos de grabación, post-producción, etc. siempre que los dueños de los derechos permitiesen hacerlo. En el fondo sería una forma más de *intertextualización*, de reuso de la producción artística.

Otra parte de la industria del cine actual es la de distribución. Al igual que le pasa a la distribución de la música, la generalización del uso de la red y la velocidad que ésta está alcanzando, mezclado con las instalación de *Home cinema* que se van extendiendo por los hogares del mundo civilizado, me temo que va a dejar a las distribuidoras en una posición muy difícil. De hecho son ellos, los estudios, las grandes distribuidoras acostumbradas a colocar sus paquetes de películas (una buena, cincuenta malas) en los cines y las televisiones las que más temen el DivX.

## Herramientas libres de producción de cine

Todo ello nos lleva también a la necesidad de herramientas libres para poder hacer todas este tipo de cosas. Por ejemplo, FreeFilm[3] es un proyecto libre con esa idea. El proyecto está en pañales y muestra signos de estar bien orientado, aunque parece demasiado ambicioso.

La mala noticia es que la industria no está por la labor. Estoy convencido de que estas herramientas van a ser perseguidas con la idea de que prohibiéndolas se eliminará la posible competencia del cine libre. La excusa como siempre

[3] http://freefilm.sourceforge.net/

será el pirateo y que estas herramientas se pueden usar para piratear contenidos protegidos. Tendremos los mismos argumentos que hemos visto en las demandas contra Jon Johansen por implementar el algoritmo (DeCSS) que permite ver los DVD legales en GNU/Linux.

Un ejemplo de esta posible persecución es el proyecto Broadcast 2000[4] del grupo Heroine Virtual[5]. Este grupo ha decidido dejar de distribuir este software bajo licencia libre que permitía editar audio y vídeo por miedo a las denuncias de los *amigos* de la RIAA.

Desde luego el primer paso que necesitamos para poder disponer de cine libre es disponer de las herramientas libres adecuadas para poderlo hacer. En particular, necesitamos herramientas básicas de edición de vídeo para los formatos usuales, herramientas para producir efectos sobre imágenes (mosaicos, ondas, etc.), drivers para los dispositivos específicos, software para crear personajes virtuales y para emplearlos, software de streaming y de multidifusión, etc. etc.

Esta parte, la construcción de las herramientas libres de edición de vídeo es un trabajo ingente y es el mayor problema en mi opinión para que no tengamos cine libre, que los programadores convencidos de las virtudes del software libre no tengan tiempo, o ganas para abordarlos o que no les dejen hacerlo.

## Para mentes inquietas

Si habéis tenido la paciencia de llegar hasta este punto del artículo, es que os interesa el asunto. En BarraPunto se repiten periódicamente discusiones sobre estos temas. En particular si buscáis términos como DivX, DVD, o *freefilm* podréis encontrar interesantes conversaciones entre lectores con más conocimientos y mejores ideas que las mías. Os esperamos allí.



---

[4] http://heroinewarrior.com/bcast2000.php3

[5] http://heroinewarrior.com/

# Y la información será libre... ¿o no?

Pedro de las Heras Quirós y Jesús M. González Barahona

Publicado originalmente en el número 45 de la revista Novática (mayo, junio de 2000)
y en el número 47 de la revista Archipiélago (junio, julio, agosto de 2001)

Las tecnologías de distribución de información están cambiando como no lo habían hecho nunca antes en la historia. Las posibilidades que nos proporcionan estos cambios y los desafíos a los que nos enfrentan son también nuevos en la historia, y tienen una potencia capaz de modificar muchos fundamentos básicos de la sociedad tal y como la hemos conocido durante los dos últimos siglos. En este ensayo tratamos por un lado de exponer la situación actual tal y como la vemos, y por otro, de dar dos visiones alternativas sobre cómo podría ser esta sociedad que nos espera. En ellas no intentamos hacer futurología, sino sólo extrapolar algunas tendencias actuales y llevarlas a lo que a fecha de hoy percibimos como sus extremos. Por supuesto, la realidad que nos encontraremos será bien diferente, y seguro que mucho más impresionante... e increíble.

## Pasado y presente

### El Software Libre: origen y situación actual

La legislación sobre patentes y derechos de copia ha marcado el desarrollo de la tecnología informática. Hasta finales de los años sesenta el software era libre. El código fuente de los programas se distribuía sin trabas entre los compradores de ordenadores como parte del servicio que recibían, para que los utilizasen libremente y sin coste adicional. En esa época, en las universidades fluía el código fuente de manera natural.

A principios de los setenta el panorama cambió drásticamente. La venta de software sin fuentes y sin permiso de redistribución ha marcado los últimos treinta años, situando entre las primeras del mundo por capitalización a empresas cuya fuente de ingresos casi exclusiva proviene de la venta de copias de software propietario. Y el caso de la industria del software no es aislado. La legislación sobre derechos de copia se ha utilizado durante varios siglos no sólo para permitir el proteccionismo en ella, sino también en otras industrias más *clásicas* (en las cuales, de hecho, tiene su origen el modelo), como la discográfica, la del vídeo y la editorial. En general, podría decirse que hasta la fecha el sector de las industrias de la información ha tratado de impedir, con éxito, el flujo libre de información con el argumento de que de esa forma la sociedad dispondrá de más y mejor información.

Por otro lado, cada vez son más las voces que reclaman una revisión de la legislación sobre patentes y derechos de copia. La posibilidad de intercambiar datos con coste prácticamente nulo gracias a Internet es, en gran parte, la razón

que está guiando este proceso de revisión que afecta a uno de los principales sectores económicos de las sociedades desarrolladas.

En el sector informático, la situación está cambiando gracias al software libre. Cabe situar el origen de este proceso de liberación a principios de los años ochenta, cuando Richard Stallman emprende el proyecto GNU. El esfuerzo pionero y visionario de Stallman y el trabajo simultáneo y continuado de muchos programadores, ha permitido que a finales de los noventa el fenómeno del software libre adquiera consistencia y sea considerado con interés por empresas y usuarios. Puede marcarse como hito histórico la liberación del código fuente del navegador de Netscape, en 1998. Desde ese momento el software libre ha irrumpido en grandes sectores la industria informática: fabricantes de hardware como Intel, Cisco o Sony utilizan software libre sobre sus procesadores. Dell, Compaq e IBM distribuyen GNU/Linux con sus equipos. Nuevas compañías cuya fuente de ingresos depende del éxito del software libre, como Red Hat Linux o VA Linux, han conseguido en el NASDAQ una financiación que hace sólo un año habría sido simplemente increíble.

Aún así, está por ver si existe un modelo económico viable que posibilite que una parte importante del software desarrollado por la industria se distribuya como software libre. Los próximos años nos mostrarán si somos o no capaces de encontrar este modelo.

## No sólo el software quiere ser libre

La distribución digital de información (audio, vídeo, libros, software) está alterando la industria tradicional. Internet ha hecho posible que cualquier persona pueda intercambiar fácilmente información digitalizada con el resto de internautas. La experiencia durante este último año con programas como Napster, que actúa como directorio de grabaciones audio en formato MP3, ha alarmado tanto a la industria del sector que ya ha emprendido acciones legales contra la empresa que lo distribuye. Cualquiera puede grabar en el disco duro de su casa una canción de un CD en un fichero en formato MP3, y a través de Napster informar de la disponibilidad de ese fichero al resto del mundo. Unos minutos después alguien puede estar escuchando esa canción a miles de kilómetros. A juzgar por el número creciente de usuarios de Napster, y salvo que pensemos que los ciudadanos no saben lo que quieren, es un hecho que son muchos los que no consideran moralmente reprobable utilizar estas herramientas.

De manera simultánea a esta tendencia, la industria está tratando de emplear un buen número de métodos técnicos y legales para impedir este proceso liberalizador: libros electrónicos intransferibles que permiten sólo un cierto número de lecturas, códigos de protección en DVDs, nueva legislación como UCITA en EEUU, o aplicación estricta de la existente, como la persecución parapolicial que realiza la BSA o la detención del programador noruego del caso DeCSS-DVD.

Todos los sectores de la industria de la información se ven afectados. Hace tan sólo unas semanas Stephen King publicó un libro electrónico con protección

anticopia que en breves horas se convirtió en el libro más distribuido en un corto espacio de tiempo de la historia de la humanidad. A los pocos días ya circulan por la red copias desprotegidas del libro.

Es notable, y como mínimo un hecho sobre el que merece la pena reflexionar, que a las primeras de cambio, en cuanto los medios técnicos lo han permitido, los ciudadanos opten en masa por copiar y dejarse copiar información, aún a sabiendas de que, por ahora, es ilegal. Y esto cuando la sociedad tiene (al menos teóricamente) una experiencia acumulada de cientos de años con la legislación de derechos de copia en el sector del libro, y de casi un siglo en los sectores de grabaciones musicales e imagen.

Podría decirse que las personas tienen una tendencia natural a compartir la información. Sólo la imposibilidad técnica y las medidas coercitivas han hecho posible que hasta ahora esta tendencia no haya podido expresarse en toda su magnitud. Y por lo tanto, la sociedad tampoco ha podido experimentar nunca con las posibilidades que proporciona el libre flujo de información (salvo en sectores concretos, y de forma parcial, como por ejemplo en el campo científico). Del enfrentamiento de estas dos fuerzas contrapuestas (por un lado las presiones para limitar el uso y distribución de la información, por otro las tendencias a usar y redistribuir información sin trabas) dependerá el futuro del software libre en particular, del acceso a la información en general, y posiblemente del mismo modelo de sociedad hacia el que nos dirigimos.

## ¿Qué futuro nos espera?

### 2010: El fin de la propiedad intelectual

Año 2010. El coste de duplicación de la información ha sido prácticamente cero desde hace una década. Desde 2005 casi todos los países desarrollados incorporaron legislación para permitir el acceso gratuito y de calidad de todos sus ciudadanos a la Red. Hacia 2008 casi la mitad de la población mundial dispone de este tipo de acceso, y gracias a los programas de coordinación internacional se espera una cobertura del 85 % de los habitantes del planeta para 2015. Junto con estas medidas, la iniciativa privada y la pública han conseguido mejorar y simplificar enormemente los medios de publicación de contenidos en la Red, hasta el punto de que cualquier persona con acceso puede hacer pública, en buenas condiciones, cualquier tipo de información (desde una novela que haya escrito a un ensayo económico o político, o una obra musical, o un escenario de realidad virtual, o un programa de asistencia al aprendizaje). La producción de información de calidad (comparable a la que a finales del siglo XX era redistribuida por editoriales de libros, estudios de cine o productoras de música) se duplica cada seis meses desde principios de siglo, y está llevando a un florecimiento de la cultura y la ciencia que deja muy atrás al impacto del Renacimiento o la Ilustración.

¿Cómo ha sido posible esta situación, si la legislación internacional ya no permite cobrar derechos de autor ni derechos por patentes? Sin duda, el impacto

mayor lo han tenido las decisiones legales de primeros de siglo. Comenzaron con tímidos movimientos de algunos países limitando los monopolios de explotación de las patentes relacionadas con la información y la biología. Continuaron con las decisiones de algunos pequeños estados de retirarse (o no incorporase) a los tratados internacionales que limitaban el libre flujo de la información entre los ciudadanos (en aquella época llamadas *leyes de propiedad intelectual*). Al principio, las presiones que tuvieron que soportar (incluidas amenazas de separación de la Red, bloqueo de intercambios de bienes culturales, etc.) fueron enormes. Pero poco a poco, estas presiones se mostraron absolutamente inoperantes frente al desarrollo de la propia Red, y a la enorme ventaja competitiva de estos estados en el mercado global de conocimientos, cultura y tecnologías de la información.

Hacia 2005, la situación para zonas económicas como la Unión Europea era claramente insostenible. Por un lado, las limitaciones al flujo libre de información les impedían mantener sus propios sectores de generación de información. Sus propios ciudadanos preferían cada vez más utilizar (y producir, mediante agentes interpuestos) información en las zonas libres. Muchos contenidos se desarrollaban cada vez más en el antiguamente llamado tercer mundo, que se estaba sumando más y más a las áreas que no controlaban el flujo de información. En 2006, la Unión Europea fue la primera zona económica del mundo desarrollado que sometió a referéndum popular su legislación sobre control del flujo de información. Tras una enconada campaña, triunfó claramente la propuesta de eliminar masivamente estos controles. Hacia 2008, el resto del mundo desarrollado se vio forzado a realizar *referendums* similares, o simplemente a abolir esa legislación.

En el campo informático, podemos afirmar que el software libre ha permitido durante la década que termina que los centros de educación y las industrias locales de muchas regiones del mundo puedan producir programas de tecnología punta, no quedándose descolgados de los desarrollos más interesantes de esta época. Aunque sea difícil de comprender hoy día, conviene recordar que a finales del siglo XX muchos de estos países prácticamente tenían vedada la participación en la industria informática más que como compradores, al no tener una industria fuerte de software propietario (el modelo imperante en esos momentos).

El hecho de que grandes proyectos de software como GNOME o la distribución de GNU/Linux Debian contasen con numerosos desarrolladores en países de Europa o Sudamérica planteó al principio de la década del 2000 interesantes reflexiones de cara al futuro. Diez años después podemos afirmar que este hecho modificó la balanza tecnológica en el sector del software, equilibrando la situación que hasta hace poco era favorable a los intereses de los EEUU.

Durante estos últimos años han surgido nuevas formas de generar recursos para hacer posible la creación de contenidos, aunque muchas de ellas tampoco son tan nuevas. En el campo del software libre ya habían emergido a finales del siglo pasado modelos de financiación alternativos, generalmente basados en la prestación de servicios alrededor del software desarrollado, o bien en el cobro por desarrollos específicos.

La pasada década ha demostrado que los supuestos que manejaban las industrias audiovisual y del libro para justificar el proteccionismo que les garantizaba la legislación de derechos de copia eran falsos. Durante estos años no ha cesado la producción de contenidos artísticos y técnicos (desde música hasta películas y libros electrónicos) como se quería hacer creer. Antes al contrario, entre el 2000 y el 2010 hemos podido conocer nuevos artistas y la variedad de contenidos ha sido superior a la que estábamos acostumbrados en el pasado siglo. Los nuevos mecanismos de financiación que se han ido descubriendo han hecho aflorar un mayor número de tendencias. Hemos asistido a la desaparición de los fenómenos de masas del siglo XX, provocados y controlados férreamente por la industria de contenidos, y a la vez hemos sido testigos de otros nuevos, emergidos del gusto de los ciudadanos. Hemos tenido, en resumen, la oportunidad de elegir libremente a quién subvencionábamos para que produjera nuestras melodías preferidas, dirigiese y/o interpretase las películas que más nos gustaban, o escribiese los libros y el software que necesitamos.

A finales de la década, la economía mundial continúa creciendo, gracias a los nuevos servicios demandados por esta sociedad de la información libre. Por primera vez en la historia, más de la mitad de la población mundial participa de este crecimiento, ya que las posibilidades de ofrecer servicios de información competitivos desde cualquier parte del mundo cada vez es más real. Los países desarrollados aún tienen cierta ventaja competitiva, debido a su mejor infraestructura de comunicaciones, pero las diferencias están reduciéndose rápidamente, ya que todos están interesados en que esta nueva sociedad de productores-consumidores de información se extienda lo más rápidamente posible a todo el planeta.

Y los cambios no han hecho más que empezar...

### 2010: La propiedad sobre todo

Mientras la sociedad seguía preocupada por la economía *tradicional* de los bienes tangibles, la legislación sobre control de la información se desarrollaba a sus espaldas. Ingentes campañas de publicidad modelaban el pensamiento de los individuos del mundo desarrollado, y estas ideas eran después exportadas al resto del planeta. Algunos países trataron de oponerse a estos cambios, por ejemplo no reconociendo patentes sobre tecnologías básicas para la cura de enfermedades. Pero la oposición combinada de los gobiernos de los países desarrollados y de las grandes *nuevas* empresas que tenían en la venta de derechos sobre la información su principal negocio hicieron que la presión sobre estos estados fuera difícil de aguantar.

Hacia 2005, prácticamente todos los estados se habían adherido (de agrado o debido a fuertes presiones) a los nuevos tratados sobre propiedad intelectual. Estos tratados eran una simple extensión a la información digital de los medios pensados para al información impresa varios siglos atrás. Pero la enorme diferencia entre las nuevas tecnologías y las disponibles dos siglos antes marcaban numerosas amenazas. Con la nueva legislación, los productores de información

pueden disponer exactamente qué puede hacer un cliente con ella después de habérsela *alquilado*. Por ejemplo, los libros electrónicos personales con control de número de lecturas hicieron posible que la información se vendiese para un sólo usuario, y que se le cobrase a éste según el número de veces que consultara la obra *vendida*. El acceso a información pública, muy dificultado por la legislación sobre responsabilidad del proveedor de información, desapareció prácticamente a partir de 2007 (incluidas instituciones como las bibliotecas públicas, que no pudieron sobrevivir a las leyes que les obligaban a pagar a los productores de información por cada lector que usaba su información). Los recientes rumores relativos a la posible prohibición de la edición en papel de libros, agravarán aún más la situación, al crear en la sociedad una dependencia total de los libros electrónicos.

Mientras que el precio de acceso a la Red se ha reducido hasta ser despreciable, incluso para los habitantes de los países menos desarrollados, el coste de acceso a la información no ha hecho más que crecer en la última década. Una nueva clase social constituida por los que pueden pagarse el acceso a información de calidad, está emergiendo como la nueva clase dirigente. Y cada vez más, la única posibilidad de entrar en ella es precisamente participar en la producción de información (normalmente como asalariado de alguno de los grandes productores de información para la Red). A pesar de las tendencias de principios de siglo, la producción de información cada vez está más concentrada, y la inmensa mayoría de la gente que participa en la Red lo hace sólo como consumidor de información *de pago*. Sólo la información que es considerada como generador potencial de ingresos es interesante para los productores que controlan la información que se pone en la Red. La situación empeora por momentos, pues las sociedades generales de autores, están presionando a los gobiernos para que sólo sus asociados puedan crear y publicar obras literarias, audiovisuales y software. Se habla de un carné de autor, que restringirá aún más las posibilidades de tener una sociedad libre. En países como España, donde ya el siglo pasado se permitió que estas sociedades cobrasen dinero por cada cinta virgen de vídeo o cada fotocopia vendida, se da como segura la aprobación de la nueva legislación.

Aunque las estadísticas difundidas por los medios oficiales indican que la producción de información de calidad es cada vez mayor, lo cierto es que se han reducido drásticamente tanto la producción de información bruta como la diversidad de esta información. Los costes de producción de una película, un programa de ordenador, o una música, cada vez son en una mayor parte costes de comercialización (hay que convencer al consumidor que pague por *echar un vistazo*). Capas sociales completas no reciben ya una instrucción adecuada porque no pueden pagar más que información limitada o de baja calidad. Muchas obras no llegan nunca al público porque no encuentran un canal de comercialización adecuado.

Los productores de información piden mayores controles contra el mercado ilegal de información, que hacia el año 2008 superó (por volumen económico esti-

mado) a los de armas y a los de estupefacientes. La población reclusa por delitos relacionados con al difusión ilegal de información en la Unión Europea superó del 50 % de la población reclusa total en el año 2009. Muchos de los famosos escritores, directores de cine y programadores que en el 2007 firmaron el Manifiesto mundial en favor de un sistema de publicación de contenidos libre y comenzaron a publicar de manera independiente, fueron perseguidos, y permanecen hoy día ocultos, publicando bajo seudónimo en el mercado ilegal. Muchos intentaron luego volver a publicar a través del sistema, pero ninguno de ellos lo consiguió, al figurar sus nombres en las listas negras de autores prohibidos.

Los recursos que los estados y las empresas productoras de información dedican a la persecución de este mercado son desde hace tiempo mayores que los dedicados a educación y sanidad, a pesar de las constantes campañas de concienciación. Uno de los últimos desarrollos en este campo permitirá controlar en tiempo real toda la información visual y de sonido reproducida por un equipo. Se espera que todos los equipos con capacidad de reproducción de la información incluyan uno de estos dispositivos para el 2012, y ya está implantada a nivel mundial la legislación que declarará ilegal en 2014 la posesión y uso de cualquier aparato reproductor que no disponga de este dispositivo, que se activa únicamente tras la identificación individual por métodos genéticos. Con él se hará por fin imposible la consulta ilegal de información por individuos que no hayan pagado por el acceso a ella.

Los expertos en economía siguen prediciendo un despegue de la economía mundial, tras los cinco años de depresión en que está sumergido el planeta después de unos años de crecimiento de principio del siglo. Pero por ahora (y a pesar de la depresión), únicamente se ha experimentado un enorme crecimiento de riqueza entre las empresas de producción de información, que siguen con grandes expectativas de crecimiento, y ya acumulan casi toda la capitalización de las bolsas mundiales de valores, en detrimento de los sectores productivos *tradicionales*, que han quedado en la práctica fuera de estos mercados.

En este año, 2010, sólo un 20 % de la población mundial tiene acceso a la Red, y por primera vez desde que existe, este año se espera que este número disminuya, ya que muchos abonados no pueden pagar las tasas privadas mínimas de acceso a la información.

## Algunas referencias

¿Son estos escenarios futuristas realmente posibles? ¿Están las cosas hoy realmente como las contamos? Desde luego, el lector tendrá su propio criterio al respecto. En caso de que quiera contrastarlo con lo que ya está ocurriendo, le proponemos aquí algunas referencias que quizás le interese consultar.

- Derechos de autor: Cerca de 300 escritores franceses se dirigen a la ministra de cultura para que sea impuesto en las bibliotecas públicas un canon por

préstamo de libros de unas 75 ptas. por cada préstamo:
http://www.el-mundo.es/diario/cultura/7N0107.html
- Los libros comprados en tiendas como eMatter sólo se pueden leer en tu PC:
http://www1.fatbrain.com/ematter/support/faq_023.asp
- Content Guard: Tecnología Xerox para evitar que los documentos se puedan copiar, y realizar un seguimiento del uso de la obra a través de Internet:
http://www.contentguard.com/overview/technology.htm
- Curso del MIT 'Ethics and Law on the Electronic Frontier'. Incluye referencias a artículos y libros sobre la información, libertad de expresión en la red, propiedad intelectual, patentes de software, control de contenidos:
http://www-swiss.ai.mit.edu/6095
- Grupo de trabajo Electronic Book Exchange: las industrias del sector de la publicación electrónica persiguen una especificación técnica para implementar protección de copyright y distribución de libros electrónicos:
http://www.ebxwg.com/
- Anuncios de tecnologías para protección de contenidos digitales:
http://www.wired.com/news/news/technology/story/21533.html
- La ley UCITA se va aprobando en varios estados de EEUU. Esta ley está diseñada por las empresas de software propietario y prohibe, entre otras muchas cosas, que se revenda el software usado, o que se haga ingeniería inversa. Permitirá por lo tanto que las empresas puedan utilizar sin miedo a ser descubiertos formatos de ficheros y protocolos secretos:
http://www.badsoftware.com/
http://www.4cite.org/ http://www.gnu.org/philosophy/ucita.html
- La industria cinematográfica de EEUU persigue a un ciudadano noruego de 15 años por desarrollar software que permite reproducir DVDs:
http://www.eff.org/IP/Video
- La industria discográfica denuncia a Napster:
http://www.mp3newswire.net/stories/napster.html
http://www.napster.org
- La industria discográfica denuncia a mp3.com: distribuidores de música en formato MP3 a través de Internet:
http://www.mp3.com/news/546.html
- Red Hat Linux: primera empresa de software libre que cotiza en bolsa:
http://barrapunto.com/articles/99/07/16/1741238.shtml
- Netscape libera el código fuente del navegador Netscape Navigator:
http://home.netscape.com/newsref/pr/newsrelease558.html

# Cuestiones económicas

Una de las confusiones más extendidas es que el software libre es gratis. Ciertamente a veces lo es, pero también es cierto que no tiene por qué serlo. Aún así, la posibilidad de redistribución que tiene el usuario hace que a la larga si es vendido sea a un coste marginal, debido a que todo aquél que lo recibe podría a su vez redistribuirlo. Visto esto, a muchos programadores les entran sudores fríos y ven en el software libre el peligro de que las empresas no obtengan ingresos por la venta de licencias, quiebren consecuentemente y no pueden hacer frente a sus salarios.

En *El software como servicio. O de cómo producir programas libres y no morir en el intento* se presenta precisamente esta problemática y por qué el software libre no es tan *peligroso* para la industria del software como puede parecer a bote pronto. El modelo de negocio alrededor del software parece concebirse preferentemente como un servicio más que como un producto. Dentro del modelo del software como servicio, el que éste sea libre le profiere posiblemente una ventaja competitiva frente a otras soluciones, como se podrá ver.

Una de las obras maestras de esta colección es sin duda *Software libre, monopolios y otras yerbas.* En una brillante exposición se demuestra la tendencia a la existencia de un producto software dominante (algo que se ha venido a llamar *monopolio de producto*). Esta situación en el mundo del software propietario significa, sin lugar a dudas, un monopolio de empresa. En el software libre, aún existiendo un producto dominante, podrán (co)existir muchas empresas compitiendo por ese mercado. Y cuanto más dominante sea y más dinero mueva, más empresas existirán, lo que hará que la competencia sea grande y, por consiguiente, sean con gran probabilidad los usuarios de ese producto los que salgan ganando. Nótese que si tuviéramos esta misma situación, pero con predominio de un producto propietario, el efecto sería más bien el contrario: es la empresa creadora del software la que sale beneficiada y la que puede imponer el precio monopolístico a su producto *exclusivo* y dictar el futuro de ese sector, muchas veces sin hacer propias las preferencias y necesidades de los usuarios.

El último ensayo de este capítulo, *La imparcialidad de los estados y la industria del software*, está dedicado a lidiar con el argumento de que los estados deben ser imparciales y no favorecer a un tipo de software sobre otro a partir de las condiciones de licencia. La tesis que defiende es que el estado ha dejado de ser imparcial desde el mismo momento en que ha promulgado unas leyes de pro-

piedad intelectual que han posibilitado ciertos comportamientos de la industria del software y más en general de la de los contenidos.

# El software como servicio. O de cómo producir programas libres y no morir en el intento

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 25, pág. 12-13, Noviembre de 2002

Cada vez que se habla del software libre, y de si es o no viable como modelo de desarrollo, surge el tema de cómo se pueden generar recursos con él. O hablando más claramente, ¿de dónde sale el dinero? O más por lo llano aún, ¿cómo van a poder pagarse sus garbanzos los desarrolladores de software libre?

Desde luego, estas preguntas no tienen una única respuesta, pero creo que sí se pueden dar al menos unas ideas que favorezcan una discusión constructiva. Y entre ellas, la más importante, la que más soluciones puede ofrecer, es la que consiste en entender el software como un servicio, no como un producto.

## De dónde sale el dinero

Cuando alguien pregunta: "¿Cómo puedo obtener recursos si me dedico a producir software libre?", probablemente olvida que hay al menos otra pregunta igual de difícil de responder: "¿Cómo puedo obtener recursos si me dedico a producir software propietario?" Porque tampoco hay una receta mágica que permita producir un buen programa propietario, poner la mano, y recibir el dinero que se pueda merecer por ello. No son, desde luego, extraños los casos donde un buen programa simplemente no tiene éxito comercial, y la empresa (o el desarrollador) que lo construyó no recupera de ninguna manera el esfuerzo invertido en él. Muchos miran las empresas de software propietario con éxito, y tienden a pensar que el asunto es bien fácil: "si a ellos les ha ido bien...". Mi consejo, en estos casos, es que miren a su alrededor. En su entorno cercano (en su ciudad, en su país incluso) ¿cuántas empresas informáticas locales tienen un modelo de negocio centrado en vender sus propios productos como software propietario? Normalmente las puedes contar con los dedos de muy pocas manos...

En el mundo del software libre, la situación es similar. Tampoco hay recetas mágicas. Y en el fondo, las cosas no son demasiado distintas. La mayor diferencia es que cuando tratas con software libre es muy difícil conseguir ingresos por venta de licencias del programa, que es en muchos casos la principal fuente en el caso del software propietario. Pero el resto de las vías de ingreso son básicamente las mismas...

## El software como producto

Las empresas informáticas que centran su modelo de negocio en la venta de licencias de uso de programas propietarios que han construido, tratan el software

como un producto. Un producto muy especial, sobre el que sólo te venden ciertos derechos, y sobre el que te impiden ejercer muchos otros (como el de copia, el de redistribución, el de mejora, etc.) Pero un producto al fin y al cabo. La relación entre el usuario y el productor es muy similar a la que tienes cuando compras un lapicero: ninguna. En la práctica, aunque a veces el usuario dispone de números de teléfono para resolver problemas, y aunque teóricamente podría tener derecho a que el fabricante le resuelva los problemas que su producto ocasione, ¿cuántas veces se usan (o sirven para algo)? En general, sólo cuando la venta incluye explícitamente condiciones de servicio. Esto es, cuando el fabricante no vende su programa como un mero producto.

Es importante darse cuenta de que detrás de la idea del software como producto está la suposición de que grupos muy grandes de usuarios tienen las mismas necesidades, y están dispuestos a adquirir un producto *estándar* que las atienda, y que nunca van a necesitar nada que no esté previsto en él (o al menos no van a estar dispuestos a pagar por cambiar lo que se les ofrece en el producto). Es la producción en masa llevada al campo de los programas: construye la infraestructura para fabricar un producto, y rentabilízala fabricando (y vendiendo) millones de ellos. Si acaso, cada pocos años, ofrece un producto mejorado, y vuélvelo a fabricar por millones.

Pero en el caso de la *factoría informática*, hay varias diferencias con los productos *tradicionales*. La fundamental, que se está tratando con algo tan dúctil como un programa. Algo que se puede modificar, adaptar o mejorar sin necesidad de grandes infraestructuras ni costosas inversiones. Cuando una empresa de automoción decide simplemente incluir un nuevo color en una de sus gamas de coches, tiene que reconvertir sus factorías, con costes no despreciables. Pero si a mí me interesa un nuevo menú en mi procesador de textos, alguien con conocimiento del programa lo puede hacer a un coste ridículo.

Si adaptar el programa al usuario es tan fácil, ¿por qué renunciar a hacerlo, cuando el usuario esté dispuesto a pagarlo? ¿Por qué va a preferir que le den *café para todos*?

## El software como servicio

Muchas empresas informáticas viven precisamente de dar servicios, que pueden incluir (o no) algunos programas informáticos desarrollados por ellos. Porque esto es lo que demandan muchos usuarios, y además estos usuarios son los que habitualmente están dispuestos a pagar más dinero. Y esta vía de ingresos para nada está vedada a las empresas que producen software libre.

La consideración del software como un servicio, además de ser mucho más cercana a los intereses del usuario, supone un cambio muy interesante de las reglas de juego, y más cuando se trabaja con software libre. Supone que la importancia se traslada del código fuente en sí mismo (el programa) al conocimiento

sobre el programa. Y esto es lo que los productores de software libre tienen que rentabilizar.

Por supuesto, sigue haciendo falta tener programas de calidad. Pero ahora se pueden realizar con mucha más facilidad, porque ya no se está obligado a construirlos completamente *dentro de la casa*. Se pueden usar componentes libres ya disponibles, o directamente reutilizar código de otros programas libres. Esto puede reducir considerablemente los costes de desarrollo. Además, es bien conocido que la fase de depuración de errores y pruebas se simplifica mucho si se sabe promover la ayuda de los usuarios.

En cualquier caso, una vez el programa esté desarrollado, el productor no podrá, en general, venderlo como producto (salvo a un precio muy bajo, pues no puede restringir la copia). Pero sí podrá venderlo como parte de un conjunto que ofrezca lo que el usuario quiera. Por ejemplo, podrá venderlo junto con una garantía de mantenimiento. O certificado para ciertas tareas. O incluyendo formación sobre su uso. En otras palabras, podrá ofrecer servicios basados en el programa. Y naturalmente, el servicio por excelencia (aunque no necesariamente el que más ingresos proporcione) será la adaptación del programa a las necesidades del cliente que esté dispuesto a pagar por ello.

## ¿Y qué gano desarrollando?

En este punto, siempre suele surgir la siguiente pregunta: "Si lo que se vende es servicio sobre el programa, y no el programa en sí mismo, ¿qué ganas desarrollándolo? ¿Por qué no dar simplemente servicio sobre programas hechos por otros? Y en ese caso, ¿quién hace esos programas?".

De nuevo, la respuesta no es simple, pero se pueden dar algunas pistas. En primer lugar, está la obviedad de que si el programa sobre el que quieres dar servicio no existe, tendrás que desarrollarlo. Éste es el caso si quieres abrir un nuevo nicho de mercado para el software libre. Y naturalmente, en esta situación serás el primero en tener un programa libre en ese nicho... Y dar el primero es dar muchas veces. Cualquier potencial competidor que quiera usar tu producto para proporcionar servicios basados en él está en desventaja contigo. Primero porque eres la fuente original del programa. A poco que lo sepas gestionar, serás el punto de referencia *fiable* para cualquier usuario. Además, si todo el desarrollo lo ha hecho tu empresa, tendrás el mejor conocimiento sobre el programa, y recuerda que es justamente de ese conocimiento del que puedes obtener ingresos. Por último, mientras controles el desarrollo del programa de forma satisfactoria para tus usuarios, tú serás el que siempre sepa con un poco de antelación por dónde van las cosas. Naturalmente tendrás que rentabilizar esta ventaja, pero las cosas están de tu parte.

En segundo lugar, puede que simplemente te interese desarrollar el programa para competir, quizás basándote en otro programa libre ya existente. En ese caso, podrás hacer el desarrollo por un coste relativamente pequeño, y diferenciarte

mucho de tu competencia. En el momento en que tu programa sea un producto claramente diferente, estás en la misma situación que he descrito en el párrafo anterior.

Y hay más escenarios. Por ejemplo, puede que lo que ocurra es que hagas un pequeño desarrollo y que un competidor lo mejore mucho haciendo un programa diferenciado. Si es mucho mejor que el tuyo, quizás te interese incorporar sus mejoras en el tuyo. Si elegiste correctamente la licencia, tu competidor se va a ver forzado a compartir sus mejoras contigo, y tú simplemente tendrás que copiarlas y adaptarlas a tu programa para posicionarte mejor.

En cualquiera de estos casos, el productor no tiene garantizado un modelo de negocio, pero tiene ventajas competitivas que puede aprovechar para conseguir la rentabilidad. Al final, como siempre en una empresa, será una combinación de buen producto, buena gestión, buena comercialización, satisfacción del cliente y suerte lo que hará que tu empresa (o tu negocio personal) prospere. Pero al menos reconocerás que hay posibilidades...

## ¿Y qué gano haciendo que el producto sea libre?

Otra pregunta interesante es ésta, "¿y por qué voy a hacer que mi programa sea libre?". Además de motivos éticos muy importantes, habitualmente esta pregunta tiene una respuesta muy simple: porque te va a ayudar a competir mejor. Y en muchos casos, aún más: porque sólo así tienes alguna oportunidad.

Por ejemplo, considera el caso de los programas ofimáticos. Imagina que desarrollas un estupendo programa, que puede hacer lo mismo que el líder en el mercado. ¿Crees que venderás muchas licencias si lo tratas de comercializar como software propietario? ¿Por qué lo va a querer el usuario en lugar del que lleva usando mucho tiempo? La única forma viable, hoy día, de entrar en ese nicho es mediante un programa libre. Puede que de esa manera tengas éxito o no, y consigas (o no) muchos ingresos. Pero si te sales del modelo de software libre, tus posibilidades son realmente mínimas.

Por supuesto, la situación puede no ser la misma en otros nichos, y tendrás que hacer un análisis cuidadoso. Y en cualquier caso, no olvides que puede que sea un competidor tuyo el que entre en tu mercado con un producto libre. Tendrás que estar en una posición muy fuerte para poder mantenerte como líder, y conservar tu fuente de ingresos...

## ¿Y ahora qué?

Para tener una panadería boyante no basta con ser un buen productor de pan. Hay que ser también un buen empresario, y aplicar todas las estrategias posibles para rentabilizar y dar publicidad a lo que se produce. El mundo del software libre no es muy diferente. Ofrece oportunidades, pero hay que saber aprovecharlas. Hasta aquí sólo he tratado de mostrar cómo la percepción del

software como servicio te puede dar una idea de cómo conseguir ingresos, y cómo eso no está reñido con dedicarlos fundamentalmente a la producción de programas. El resto, desde luego, es mucho más complejo.

Pero si te animas a explorar este camino, no olvides que no eres el primero que lo hace, y que hay empresas que sobreviven en un mundo tan complejo como el de las tecnologías de la información desde hace años dedicándose a producir, mantener y dar servicios basados en software libre. Analizando cómo funcionan, se puede aprender mucho. Aunque claro, siempre hay que considerar la posibilidad de probar nuevas ideas, y de descubrir un nuevo modelo de negocio... Ocurre pocas veces, pero ocurre.

# Software libre, monopolios y otras yerbas

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 3, pág. 12-13, Noviembre de 2000

El mercado informático tiende al monopolio de producto en todos sus ámbitos. Los usuarios quieren rentabilizar el esfuerzo realizado en aprender cómo funciona un programa, las empresas quieren encontrar gente formada en el uso de su software, y todos quieren que los datos que gestionan puedan ser entendidos por los programas de las empresas y personas con las que se relacionan. Por eso cualquier iniciativa dedicada a romper una situación de facto donde un producto domina claramente el mercado está destinada a producir más de lo mismo: si tiene éxito, vendrá otro producto a ocupar ese hueco, y en breve tendremos un nuevo monopolio. Sólo los cambios tecnológicos producen, durante un tiempo, la inestabilidad suficiente como para que nadie domine claramente.

Pero la situación donde un producto domina el mercado hasta el punto de constituirse en un monopolio de facto no es necesariamente indeseable. En realidad, lo preocupante es lo que conlleva: cuando un producto domina el mercado, sólo hay una empresa que lo controle. El software libre ofrece una alternativa a esta situación: los productos libres pueden estar promovidos por una empresa en concreto, pero esta empresa no los controla, o al menos no hasta los extremos a los que nos tiene acostumbrados el software propietario. En el mundo del software libre, un monopolio de producto no implica necesariamente un monopolio de empresa. Por el contrario, sea el que sea el producto que domine el mercado, muchas empresas pueden competir en proporcionarlo, mejorarlo, adaptarlo a las necesidades de sus clientes y ofrecer servicios alrededor de él.

## Elementos que favorecen el monopolio de producto

En informática es muy común que haya un producto claramente dominante en cada segmento de mercado. Y eso es normal por varios motivos, entre los que cabe destacar los siguientes:

- **Formatos de datos**. En muchos casos el formato de datos está fuertemente ligado a una aplicación. Cuando un número suficientemente alto de gente la usa, su formato de datos se convierte en estándar de facto, y las presiones para usarlo (y la aplicación por tanto) son formidables.
- **Cadenas de distribución**. Normalmente uno de los problemas para empezar a usar un programa es obtener una copia de él. Y normalmente es difícil encontrar los programas que no son líderes en su mercado. Las cadenas de distribución son costosas de mantener, de forma que los competidores minoritarios lo tienen difícil para llegar a la tienda de informática, donde el usuario

final pueda comprarlos. El producto dominante, sin embargo, lo tiene fácil: el primer interesado en tenerlo va a ser la propia tienda de informática.

- **Márketing**. El márketing *gratuito* que obtiene un producto una vez que lo usa una fracción significativa de una población determinada es enorme. El boca a boca funciona mucho, también el preguntar e intercambiar información con los conocidos. Pero sobre todo el impacto en los medios es muy grande: las revistas de informática hablarán una y otra vez de un producto si parece ser el que más se usa. Habrá cursos de formación para él, libros que lo describan, entrevistas a sus usuarios, etc.
- **Inversión en formación**. Una vez se han invertido tiempo y recursos en aprender cómo funciona una herramienta, se está muy motivado para no cambiar. Además, usualmente esa herramienta es la que ya domina el mercado, porque es más fácil encontrar personal y material que ayuden a aprender a usarla.
- **Software preinstalado**. Recibir una máquina con software ya instalado desde luego es un gran incentivo para usarlo, incluso si hay que pagar por él aparte. Y normalmente, el tipo de software que el vendedor de la máquina va a estar dispuesto a preinstalar será solamente el más utilizado.

Hay otros motivos que favorecen el monopolio de producto. Con esta lista sólo quería dejar claro que hay elementos fuertes, y muy difíciles de contrarrestar, que hacen que en el campo de software aparezcan de forma natural estos monopolios.

## La situación actual: monopolios de empresa

En el mundo del software propietario un monopolio de producto en un segmento cualquiera equivale a un monopolio por parte de la empresa que lo produce. Por ejemplo, tenemos estas situaciones monopolísticas de facto (o casi) de producto y empresa en los mercados de sistemas operativos, autoedición, bases de datos, diseño gráfico, procesadores de textos, hojas de cálculo, etc.

Y esto es así porque la empresa en cuestión tiene un gran control sobre el producto líder. Tan grande que sólo ellos pueden marcar la evolución del producto, las líneas fundamentales en las que se va a desarrollar, su calidad, etc. Los usuarios tienen muy poco control, dado que estarán muy poco motivados para probar otros productos (por los motivos que se han comentado en el apartado anterior). Y naturalmente, las empresas competidoras poco podrán hacer, salvo tratar de desafiar la posición dominante del producto mejorando excepcionalmente los suyos (para tratar de contrarrestar esos mismos motivos), normalmente con poco éxito.

Esta situación pone a todo el sector en manos de la estrategia de la empresa dominante. Todos los actores dependen de ella, e incluso el desarrollo de la tecnología software en ese campo estará mediatizada por las mejoras que le haga a su producto. En el caso general, ésta es una situación donde aparecen los peores efectos económicos del monopolio, y en particular, la falta de motivación de la

empresa líder para acercar el producto a las necesidades (siempre en evolución) de sus clientes. Éstos se han convertido en un mercado cautivo.

## Software libre: sólo monopolios de producto

Sin embargo, en el caso del software libre un monopolio de producto no se traduce automáticamente en un monopolio de empresa. Si el producto es libre, cualquier empresa puede trabajar con él, mejorarlo, adaptarlo a las necesidades de un cliente y en general, ayudar en su evolución. En otras palabras: en el mundo del software libre habrá, por los motivos discutidos anteriormente, monopolios del producto. Pero en el caso de que esto ocurra, habrá también muchas empresas interesadas en mejorar y adaptar ese producto, precisamente por su posición dominante en el mercado. Si el productor *original* (la empresa que desarrolló originalmente el producto) quiere permanecer en el negocio ha de competir con todas ellas, y por eso estará muy motivado para hacer evolucionar el producto precisamente en la línea que sus usuarios quieran. Naturalmente, tendrá la ventaja de un mejor conocimiento del programa, pero eso es todo. Tiene que competir por cada cliente.

El monopolio de producto se traduce en el mundo del software libre, por lo tanto, en competencia feroz entre empresas. Y con ello los usuarios retoman el control: las empresas en competencia no pueden más que hacerles caso si quieren sobrevivir. Y precisamente esto es lo que asegurará que el producto mejore.

## Algunos casos que ya han sucedido

Examinando la evolución de los proyectos de software libre pueden encontrarse varios casos donde esto ya ha ocurrido. Por ejemplo:

- **Apache** es desde hace tiempo líder en el mercado de servidores de web. Pero hay muchas empresas que están detrás de Apache, desde algunas muy grandes (como IBM) a otras muy pequeñas. Y todas ellas no tienen más remedio que competir mejorándolo, y normalmente contribuyendo al proyecto con sus mejoras. A pesar de que Apache es casi un monopolio en muchos ámbitos (por ejemplo, es casi el único servidor web que se considera sobre la plataforma GNU/Linux o *BSD), no depende de una sola empresa, sino de literalmente decenas de ellas.
- **Las distribuciones de GNU/Linux** son también un caso interesante. GNU/Linux no es desde luego un monopolio, pero es posiblemente la segunda opción en el mercado de sistemas operativos. Y eso no ha forzado la situación donde una empresa tenga su control. Al contrario, hay decenas de distribuciones, realizadas por empresas diferentes, que compiten libremente en el mercado. Cada una de ellas trata de ofrecer mejoras que sus competidores tienen que adoptar a riesgo de ser echados del mercado. Pero además no pueden separarse

demasiado de lo que es GNU/Linux *estándar*, pues eso es rechazado por los usuarios como una *salida de la norma*. La situación después de varios años de crecimiento de la cuota de mercado de GNU/Linux nos muestra a decenas de empresas compitiendo y haciendo evolucionar el sistema. Y de nuevo, todas ellas están detrás de satisfacer las necesidades de sus usuarios. Sólo así pueden mantenerse en el mercado.

- **GCC** es un monopolio de facto en el mundo de compiladores de C y C++ para el mercado GNU/Linux. Y sin embargo, eso no ha llevado a ninguna situación de monopolio de empresa, incluso cuando Cygnus (hoy Red Hat Linux) se ha encargado durante mucho tiempo de coordinar su desarrollo. Hay muchas empresas que hacen mejoras al sistema, y todas ellas compiten, cada una en su nicho específico, por satisfacer las demandas de sus usuarios. De hecho, cuando alguna empresa u organización específica ha fallado en el trabajo de coordinación (o así lo ha percibido una parte de los usuarios) ha habido espacio para un *fork* (división) del proyecto, con dos productos en paralelo durante un tiempo, hasta que eventualmente han vuelto a unirse (como está ocurriendo ahora para GCC 3.x).

Según vaya entrando el software libre en nuevos mercados, veremos más y más situaciones donde habrá un monopolio de facto por parte de un producto libre. Y en cada uno de esos casos veremos cómo eso no se traslada (al menos no fácilmente) en un monopolio de empresa, sino por el contrario, en una constelación de empresas pugnando por satisfacer las necesidades del usuario... por la cuenta que les tiene.

Ésta es la clave de que si en algún momento GNU/Linux se convirtiera en el líder del mercado de sistemas operativos no tendríamos *más de lo mismo*. No tendríamos una nueva situación de monopolio, con un mero cambio de nombres. Tendríamos más bien una situación donde muchas empresas se verían atraídas al mundo de GNU/Linux, ya que en él estaría la gran masa de usuarios, y por lo tanto su dinero. Para poder mantenerse en el mercado, todas estas empresas estarían obligadas a mejorar el producto (GNU/Linux), compitiendo entre sí y luchando por hacer caso al usuario.

## Y el futuro dirá

Desde luego es pronto para saber si en el futuro habrá o no muchos productos libres que se conviertan en monopolios de facto. Pero podemos estar seguros de que en los casos en los que esto ocurra no habrá que temer la posición dominante de empresas, o al menos no de la forma en que la tememos hoy día en el mundo del software propietario. Los usuarios recuperarán el control, y las empresas que quieran mantenerse en el negocio estarán ferozmente obligadas a satisfacer sus necesidades. Y quizás finalmente tengamos un escenario en el que una las ventajas de tener productos que monopolicen un mercado (volumen de mercado, rentabilización de las inversiones en formación, etc.) se combinen con

las ventajas de tener competencia entre las empresas que sirven ese mercado (y que fundamentalmente se traducen en seguir con más atención las necesidades del usuario).

Dicho de otra forma, y teniendo en cuenta que el monopolio de producto es inevitable por el propio funcionamiento de la industria del software, ¿podremos tener mercados en competencia, como en otras industrias, gracias al software libre?

# La imparcialidad de los estados y la industria del software

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 22, pág. 12-13, Julio de 2002

Últimamente estamos viendo algunas iniciativas de promoción del uso de software libre por parte de algunas administraciones públicas. Esto ha ocurrido, por ejemplo, en Francia, Alemania, Finlandia, Perú, Colombia, España y Corea. En unos casos se han quedado en simples propuestas, en otros han avanzado hasta convertirse en recomendación, ley, plan de implantación, o cualquier otra cosa. Pero en general estas propuestas han provocado un debate sobre si el estado, o cualquier administración pública, debe o no promover el software libre. Se habla de libre competencia, de que el mercado decida, de que el estado no debe favorecer a unas empresas en lugar de otras, etc., etc.

Después de varios meses de seguir (y participar en) estas polémicas, me ha dado por contemplar el asunto desde otro ángulo, poniéndolo en un contexto más amplio. En estas notas os cuento lo que he visto desde ahí.

## De cómo los estados no son imparciales

Uno tiende a pensar que los estados son fundamentalmente imparciales con respecto al funcionamiento del sector de software. A primera vista es un mercado libre, en el que cualquier empresa o particular puede comprar y vender programas. Aparentemente, los precios los fija libremente el propio mercado, y salvo algunos *detalles* de los que hablaremos más tarde, los estados no parecen dar ventajas a unos modelos de negocio frente a otros, ni a unas empresas frente a otras.

Sin embargo, cuando uno analiza la situación con más cuidado, esta supuesta imparcialidad queda bastante maltrecha. Porque son los estados los que están dictando el modelo de negocio a la industria del software, o al menos están dando ventajas muy grandes a ciertos modelos de negocio frente a otros. Y al hacerlo, están marcando también la estructura del sector informático y las características de las empresas que mejor pueden desenvolverse en él.

¿Cómo ocurre esto? Entre todas las funciones que desarrollan los estados, hay una que es muy importante: la legislativa. No olvidemos que son los estados los que dictan (por medios democráticos o no) las reglas del juego que todos jugamos, y los que se encargan luego de hacerlas cumplir.

¿Cómo ejercen los estados esta función en el caso del software? Fundamentalmente, reglamentando lo que normalmente se denomina *propiedad intelectual* (derechos de autor) y en algunos casos, también la *propiedad industrial* (patentes). Si esta legislación no existiera, los usuarios podríamos copiar libremente los

programas que recibiésemos, y en general, disponer de los ellos como nos pareciese conveniente, sin más limitaciones. No tenemos posibilidad de hacer lo que queramos con un programa propietario que recibamos sólo porque los estados han legislado que sus autores o productores puedan prohibirnos total o parcialmente hacer este tipo de cosas.

Para evitar malos entendidos, es importante también darse cuenta de que la situación es muy diferente de la que afecta a la propiedad de los bienes materiales. Si compras un pastel, o te lo comes tú o se lo das (o vendes) a otro. Pero no puedes comértelo y regalarlo (o venderlo) a la vez. Por eso, la propiedad privada de los bienes materiales es un asunto, y la de la información es otro bien distinto. Tú puedes recibir un programa, copiarlo y regalárselo (o vendérselo) a alguien, y después, tranquilamente, seguir usándolo. Como el coste de la copia es hoy día ridículo, en la práctica podemos ignorarlo en la mayoría de las situaciones.

Haya o no legislación que favorezca a la propiedad privada de los bienes físicos, quien los use no estará muy contento si otro trata de usarlos en su lugar sin darle nada a cambio. El estado aquí regula una situación que en caso contrario puede dar lugar a conflictos, ya que hay intereses encontrados. Pero esto no ocurre en las transacciones relacionadas con la información, o al menos no de la misma manera. Por ello, el estado no está regulando sobre una situación conflictiva cuando nos prohibe, por ejemplo, la copia de un programa. Simplemente, está poniendo una reglas que favorecen un determinado modelo de industria.

## Y las cosas podrían ser de otra forma...

Una vez nos hemos dado cuenta de que la situación actual es la que es solamente porque los estados han legislado en ese sentido, podemos pensar que, si la legislación hubiera ido por otros caminos, las cosas serían bien diferentes. El mundo podría ser de otra forma, y la industria del software también. Por ejemplo, no habría sido tan extraño que los estados hubieran decidido que para comercializar un programa fuera obligatorio dejar el código fuente a disposición de los clientes, y darles permiso explícito para corregir los errores que pudiera tener. Como se hace por ejemplo en el mercado inmobiliario, donde quien compra una vivienda tiene derecho a un plano donde se indique la situación de las canalizaciones, el cableado eléctrico, etc. Y naturalmente cualquiera puede hacer en su casa las modificaciones (no estructurales) que quiera, sin tener que pedir permiso al vendedor.

O podría haberse decidido que las empresas compitieran en los servicios, y por tanto no haber restringido el derecho de copia a los usuarios. En ese caso, el estado habría podido decidir que para comercializar un programa sería condición indispensable proporcionar su código fuente a todo el que lo pidiera, y no poner ningún impedimento a la redistribución. ¿Te suena este modelo? Efectivamente, los estados podrían haber decidido (podrían decidir dentro de un tiempo) que

para comercializar un programa, este tuviera que cumplir condiciones similares a las que cumple un programa libre.

Visto desde este punto de vista, las legislaciones actuales difícilmente pueden considerarse como imparciales con respecto al modelo de negocio. De hecho, el modelo de negocio tradicional en el mundo del software-producto, la venta de licencias de uso limitadas (lo que adquirimos cuando *compramos* un programa propietario) sólo es posible porque se ha legislado de forma que lo sea. Y al hacerlo, no sólo el estado ha decidido que éste sea el modelo más *popular*, sino que ha hecho posibles consecuencias como que cuanto mayor sea el mercado donde se venden licencias, más beneficios se consigan (de forma casi directa: los costes aumentan poco, los ingresos mucho), o que haya empresas que detenten monopolios *de facto* en amplios nichos (los usuarios prefieren usar el producto mayoritario, nadie más que la empresa que lo produce puede proporcionárselo, y nadie puede competir proporcionando el mismo producto).

## Aún hay más

La legislación sobre propiedad intelectual no es la única forma en que los estados están influyendo en la industria del software, o ayudando a que las empresas con éxito en esta industria sean de cierta forma. Un ejemplo, nada menos que del Foro de la Convención Europea, la institución que tiene el encargo de proponer una Constitución Europea[1]:

> "Aunque formalmente sea posible enviar estos documentos por correo o fax a la Secretaría de la Convención, sólo podrán tratarse efectivamente en el marco del Foro, publicarse y difundirse los textos enviados por vía electrónica (se recomienda el formato Word)."

He escogido este ejemplo por lo relevante de la actividad a la que se refiere, pero textos muy similares se podrían haber encontrado en cientos de otros lugares relacionados directamente con entes dependientes de estados.

¿No constituye esta nota una forma de favorecer descaradamente a un producto concreto de una empresa concreta, frente a todos sus competidores? Al menos en este caso nos dicen solamente *se recomienda*. En otros casos nos encontramos con la simple y llana obligación... Como se puede ver, estamos llegando a un momento donde para poder participar en la construcción europea, en la elaboración de lo que podría llegar a ser nuestra Constitución Europea necesitamos tener una licencia de Microsoft Word...

Y desgraciadamente éste no es un caso aislado. En muchos casos, para comunicarnos con nuestras administraciones públicas (a nivel local, regional, estatal o supranacional) tenemos que usar determinados productos de determinadas marcas. Para pagar impuestos, para acceder a ayudas de investigación, para poder

[1] Tomado de http://europa.eu.int/futurum/forum_convention/how_es.htm

participar en subvenciones a la compra de material informático, etc., etc. La lista parece no tener fin.

¿Puede decirse que éstas son prácticas imparciales? Las administraciones públicas, el estado en fin, están promoviendo, conscientemente o no, el uso de ciertos productos, ayudando directamente a engrosar las cuentas de beneficios de ciertas empresas, y no de otras.

Y hay otros muchos casos, como cuando con dinero público se pagan cursos de formación, supuestamente de ofimática, que se convierten en realidad en clases de capacitación para el uso de ciertas herramientas de software propietario. O cuando se malgastan cantidades ingentes de recursos públicos en programas de investigación y desarrollo destinados a subvencionar la creación de programas propietarios, que sólo podrán comercializar, y convertir en beneficios, los mismos que reciben las subvenciones. Y más, y más y más.

## ¿Hay que darle la vuelta a la tortilla?

Por lo tanto, el estado dista mucho de ser imparcial con respecto a la industria informática. Ha regulado el sector, vía legislación sobre propiedad intelectual, definiendo así tanto el modelo de negocio fundamental que se emplea en él como el tipo de empresa que mejor puede competir con esas reglas de juego. Como consecuencia, también ha definido las prácticas *dominantes*: ocultación de información técnica, prácticas que favorecen monopolios empresariales, etc. Por si fuera poco, favorece directamente a ciertas empresas, a veces de forma inconsciente, en otras con clara intencionalidad.

A pesar de que las reglas del juego son tan poco neutras, el software libre ha mostrado que hay (al menos) otra forma de hacer las cosas. Y que esa otra forma es capaz de generar software de suficiente calidad, y en suficiente cantidad, como para satisfacer las necesidades de un grupo creciente de usuarios. Por lo tanto, ha llegado el momento de plantearse una pregunta que, lejos de ser académica, tiene un directo impacto práctico: "¿Qué modelo de sector informático tiene que favorecer el estado?" Si respondemos a esta pregunta, el tipo legislación a promover aparecerá como consecuencia.

Naturalmente, responder esta pregunta no es fácil. Pero hay que trabajar en encontrar esa respuesta, porque en cualquier caso, al imponer la situación actual, el estado ya la está respondiendo, y quizás no de forma óptima.

Desde mi particular punto de vista, la respuesta ha de estar basada en la maximización del bien común. Esto es, debemos tener el modelo de sector informático que mejor software produzca para la sociedad. Concretar qué es *mejor* es sin duda más difícil. Pero una primera aproximación podría ser *con menos recursos, con más calidad, con más cantidad, llegando a todos los usuarios*. A la luz de estos criterios, o de otros similares, es como el estado debe estudiar la conveniencia o no de un modelo u otro. Y a partir de las conclusiones de ese estudio, legislar de forma que sea ese el modelo favorecido.

Hoy, sin embargo, el estado nos está imponiendo una situación donde se prima un determinado modelo de producción de software sin haber discutido ampliamente si eso es lo que consideramos mejor. Por lo tanto, la discusión no es si el estado debe dejar de ser imparcial para pasar a promover un modelo dado, sino si, siendo parcial, como ya es, debe primar el modelo que está primando ahora, o debe dar señales claras de que hay otros modelos más interesantes para la sociedad.

Esto es lo que se está empezando a hacer, de forma muy tímida, en algunas partes del mundo. Y esto es lo que está siendo ferozmente criticado por los beneficiados por la situación actual. Porque cuando se habla de que el estado no debe inmiscuirse ni regular una industria como la informática, lo que se está diciendo en realidad es que el estado debe seguir con la estricta regulación que ha puesto en vigor hasta el momento. Que debe seguir inmiscuyéndose en cómo funciona el sector informático, pero sólo para promover aún más el modelo actual.

En mi opinión, lo que tiene que ocurrir es lo contrario: que se deje de promover ciegamente el modelo actual, que se estudie cuál es que queremos, y que se pase a promover ése, sea el que sea. Y entre todos los que se nos puedan ocurrir, propongo que se estudie como candidato a promover el modelo del software libre. Y por eso animo a los estados que ya han comenzado tímidamente ese camino a que continúen por él: el sitio al que lleva difícilmente puede ser peor que en el que estamos ahora.

# El lado más técnico

Antes de que el lector algo *alérgico* a las tecnologías de la información se salte los ensayos contenidos en este capítulo por considerarlos que van a estar fuera de su alcance, una advertencia: no se asuste; el lobo no es tan fiero como lo pintan.

Y es que es cierto que los artículos que se va a encontrar en las siguientes páginas tienen un cierto caracter más pragmático y no tan filosófico como los anteriores, pero sin lugar a dudas muestran los beneficios del software libre desde una perspectiva ajustada al perfil de ciudadano de a pie. Así que, deles una oportunidad. Puede resultar interesante entender qué es lo que lleva a mucha gente a participar en el desarrollo de software libre.

*¿Y cómo hago para que mi código sea libre?* es una rápida guía para hacer un proyecto de software libre. Hasta aquí, nada difícil: para tener un proyecto libre, sólo hace falta ponerle una licencia libre y publicarlo (en algún sitio de Internet, por ejemplo). Pero el software libre, probablemente no ha tenido tanto éxito por eso (al menos, no sólo por eso). Los proyectos que han causado furor han ido mucho más allá y han hecho uso de una estrategia para maximizar la captación de desarrolladores y usuarios. Tanto es así, que podríamos decir que este artículo trata más bien de cómo conseguir una comunidad que haga del proyecto inicial un trabajo colaborativo fruto de muchas mentes conectadas en red.

*Con todo al aire* es el singular título del siguiente ensayo, que se centra en el hecho de que en el software libre no sólo el código fuente es público, sino también muchos otros elementos y procesos de su desarrollo (las discusiones en las listas de correo, los sistemas de notificación de erratas, las propias erratas, etc.). Este hecho parece ir en contra de una de las más asentadas prácticas en el mundo de la empresa moderna (incluso de gobiernos democráticos) coloquialmente conocida como *los trapos sucios se lavan en casa*. Aún así, resultar ser uno de los pilares básicos del software libre y por eso se analizarán los pros y los contras de la transparencia en el desarrollo de software.

*KDE o GNOME, ¿es ésa la cuestión?, ¿es la cuestión GNOME o KDE?* aborda la sana competencia que existe entre los escritorios de GNOME y de KDE. Estos proyectos, orientados a conseguir una interfaz gráfica de usuario para GNU/Linux, son tema de amplio (y a veces acalorado) debate entre sus respectivos seguidores en los foros de discusión sobre software libre. Por si cada uno no tuviera ya su opinión, este artículo se atreve a añadir una más.

# ¿Y cómo hago para que mi código sea libre?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 30, pág. 12-13, Marzo de 2003

"Tengo un programa, que escribí por tal y cual razón, y me gustaría que fuera libre, ¿qué tengo que hacer?" Ésta es una pregunta bastante habitual en estos días, en los que tantos programadores (especialmente los que gustan de programar) se dan cuenta de lo poco que pierden y lo mucho que ganan haciendo que sus *juguetes* sean libres.

Como siempre, hay al menos dos respuestas para ella: la corta y la larga. La primera es bien simple: escoge una licencia libre, y publica tu programa acogiéndote a ella. La segunda es bastante más complicada, y un tanto poco habitual. Este artículo trata de resumirla en unas pocas páginas (la respuesta completa probablemente llenaría al menos un libro).

## Comenzar por el principio

Vale. Ya has decidido que tu programa estaría mucho mejor siendo libre. Ese amigo pesado te ha acabado de convencer, o quizás tú mismo has puesto en la balanza los pros y los contras y has decidido que lo mejor es liberarlo. ¿Por dónde empiezas ahora?

Puede parecer obvio, pero para comenzar, hay que asegurarse de que uno es el autor único de software que se quiere liberar (o si no es así, que todos los autores están de acuerdo con los términos de la liberación). Esta parte no es muy diferente de su equivalente en el mundo del software propietario: sólo se puede comercializar un programa si todos sus autores están de acuerdo en ello, o han cedido sus derechos a quien lo comercialice. Si el proyecto ha empezado de cero, y es relativamente joven, esto suele estar muy claro. Pero no siempre es así. Especialmente en el mundo del software libre, pueden haberse recibido contribuciones de alguien que no conocemos, o quizás los términos en los que colaboró cierto programador están algo oscuros. Es muy importante aclarar todos esos extremos antes de liberar, si no queremos arriesgarnos a problemas futuros.

Una vez resuelto el problema de la autoría, debemos elegir, como indicaba la respuesta corta, la licencia. Según el enfoque de los autores, según sus intenciones prácticas, según sus motivos para hacer la liberación, se puede elegir entre una gran cantidad de licencias libres (puedes echar por ejemplo un vistazo la lista de licencias libres de la Free Software Foundation[1] o a las licencias aprobadas por la Open Source Initiative[2]). E incluso crear la tuya propia. En general, cuando

[1] http://www.fsf.org/licenses/license-list.html

[2] http://opensource.org/licenses/index.php

me preguntan, en esta fase hago dos recomendaciones: no escribir una licencia propia (necesitas un buen consejo legal para asegurarte de su validez, y que consigue los fines que persigues), y en la medida de lo posible escoger una licencia popular (que maximizará la probabilidad de que los usuarios y colaboradores potenciales la entiendan, y por tanto sepan a qué atenerse). Normalmente, la decisión fundamental será entre una licencia minimalista (que deja al receptor del programa hacer casi cualquier cosa con él, incluido redistribuirlo como software propietario) y una licencia copyleft (que transfiere a quien recibe cualquier trabajo derivado del nuestro las mismas libertades de redistribución y modificación que nosotros hemos dado), como la GPL.

Con la licencia elegida, podemos empezar el proceso. Habitualmente, se marcarán con ella todas las cabeceras del código fuente del programa, y se incluirán notas informando sobre ella visibles en tiempo de ejecución (por ejemplo, visibles en el apartado *Acerca de* de nuestro programa). Algunas licencias, como la GPL, incluso indican qué ha de hacer quien quiera usarla.

Siguiendo lo que antes llamamos *respuesta corta*, el proceso terminaría empaquetando el binario y los fuentes (por favor, no olvides los fuentes), y subiéndolo a algún sitio de Internet. Pero en realidad, si tienes ganas de hacerlo bien, no has hecho más que empezar.

## Infraestructura mínima

Si quieres que tu programa pueda desarrollarse según los modelos habituales del mundo del software libre (consiguiendo colaboración externa, publicando versiones frecuentemente, decidiendo de forma abierta sobre la evolución del código fuente, etc.) conviene que montes una infraestructura mínima. Su objetivo será doble: conseguir visibilidad para ti como autor (ofreciendo un buen sitio con información relacionada con el programa) y facilitar la colaboración de otros desarrolladores (incluyendo las colaboraciones ocasionales, como por ejemplo los usuarios que informan de errores). Hoy día, esta infraestructura consiste habitualmente en un conjunto de herramientas e información accesible vía web. Por ejemplo, te interesará ofrecer al menos:

- Una presentación del proyecto. Explicando qué hace el programa, en qué estado está, qué ideas hay para su desarrollo, etc. También es conveniente dar créditos a los colaboradores principales, y dejar claro que el proyecto admite colaboraciones (si es el caso, claro).
- Por supuesto, habrá páginas que permitan descargar el software (fuentes y binarios). Estas páginas serán normalmente muy visibles desde la entrada al sitio, ya que es una de las actividades más habituales para los recién llegados.
- Documentación y comunicados. Muchos usuarios y desarrolladores acudirán al sitio buscando información detallada sobre el programa, y sobre sus últimas novedades. Si es posible, deberían encontrarla rápidamente... Por supuesto,

aquí se pueden incluir contribuciones de usuarios (trucos de uso, guías de instalación, capturas de pantalla, etc.)

- Espacio para desarrolladores. Como mínimo con posibilidades de subir (y hacer seguimiento) de informes de error, parches, acceso a versiones no estables, etc.

En el fondo, y en la medida de lo posible, el sitio del proyecto tratará de crear a su alrededor una comunidad. Cuanto mayor y más estable sea ésta, más fácil será conseguir la masa crítica de usuarios y desarrolladores que aseguren el desarrollo futuro del programa.

## Pónselo fácil al usuario

Además de preparar bien el sitio del proyecto, es importante que cuando un usuario potencial lo visite sea fácil convertirse en usuario real. Por ejemplo, además de proporcionarle los fuentes con instrucciones claras y lo más sencillas posibles sobre su compilación e instalación, estará muy bien proporcionarle binarios listos para instalar. E incluso versiones del programa ya empaquetadas para las distribuciones más populares (lo que además facilitará su inclusión en esas distribuciones, claro).

Además, el usuario debería poder instalarse fácilmente en su propio ordenador la documentación disponible (para poder acceder a ella cuando no esté conectado). Los foros con información para novatos, las referencias a experiencias de otros usuarios, las listas de correo donde pueda hacer preguntas, etc. ayudarán a que el usuario saque rápidamente partido de tu programa.

En general, procura dar facilidades al usuario. Ten en cuenta que en muchos casos la primera impresión que se llevará del proyecto será el sitio web. Si puede encontrar en él rápidamente lo que busque, si la impresión general es buena, ya tienes dado un primer paso muy importante. Luego, cuida la instalación. Si es posible, que no tenga que hacer nada salvo lo habitual en su distribución. Si el usuario llega a ese punto, el resto ya puedes dejarlo en manos de tu maravilloso programa...

## Ya que estás en ello, aprovéchate

Cuando tengas listo el sitio web, y la primera versión de tu programa que quieras liberar, haz todo el ruido que puedas. Aprovéchate de los medios de difusión que la comunidad del software libre pone a tu disposición. Por ejemplo, sube un anuncio a FreshMeat. Avisa en las listas de novedades y en las especializadas, según el tipo de programa que hayas hecho. Y ofrece en tu sitio listas donde cualquier interesado pueda recibir a partir de ese momento novedades sobre el programa.

Como autor de un programa libre, hay más recursos de los que te puedes beneficiar, pues están puestos a disposición de los desarrolladores de esta comunidad. Los más significativos son sitios como SourceForge, Savannah o BerliOS, que proporcionan una infraestructura genérica para el desarrollo de proyectos, muy similar a (y más completa que) la expuesta en el apartado anterior. Además, su uso también servirá en cierta medida de medida de promoción, ya que sus servicios de búsqueda de proyectos son cada vez más usados cuando se quiere encontrar un software dado.

## Y esto es sólo el principio

Liberar un programa puede ser tan simple como ponerlo en la red, o tan complejo como quieras, si tratas de hacerlo con el mayor impacto posible, y beneficiándote lo más que puedas de ello. Pero lo hagas como lo hagas, es sólo el principio. Es posible que a pesar de tus esfuerzos no logres interesar a nadie con tu fabuloso programa. O quizás todo lo contrario: puede que te sorprendas de la gran expectación que tu *juguete* causa ahí fuera. En este caso, prepárate para satisfacer a tus nuevos usuarios, y quien sabe si incluso a desarrolladores que ilusionados por tu trabajo, estén más que dispuestos a contribuir con sus ideas, su tiempo y su trabajo a tu proyecto.

Ten en cuenta que la *gestión* de un proyecto libre no es algo que salga gratis, ni en tiempo ni en recursos, y que en el largo plazo tendrás que tenerla muy en cuenta, si quieres que el proyecto sea exitoso. Hace falta unas habilidades especiales, habitualmente bastante distintas de las del *programador solitario* para saber motivar, atender y satisfacer a un grupo de colaboradores voluntarios, y a una buena cantidad de usuarios. En cualquier caso, hay cosas que se pueden aprender al respecto, pero eso queda ya para otro artículo.

Como último comentario sobre la liberación de tu proyecto, ten en cuenta que si tienes suerte, y sabes gestionarla, puede que veas florecer las más puras esencias del mundo del software libre a tu alrededor... Y de todas formas, lo que es seguro es que de todas formas merecerá la pena. ¿No crees?

# Con todo al aire

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 31, pág. 12-13, Abril de 2003

Una de las características diferenciadoras del software libre es que su código fuente está al alcance de cualquiera que desee leerlo. Pero ésta no es, ni mucho menos, la única información pública que mantiene habitualmente un proyecto de software libre. También suele haber listas de correo electrónico, sistemas de gestión de errores, documentación en páginas web, etc., etc. Esta *apertura*, este esfuerzo porque gran parte de la información relacionada con el proyecto sea pública, sorprende a muchos, y especialmente cuando se compara con la situación en el mundo del software propietario, donde no sólo el código fuente suele ser un secreto muy bien guardado, sino que se esconden los errores del programa o no hay detalles sobre su arquitectura o las decisiones de diseño que le afectan.

¿Por qué en el software libre es habitual poner tanta información a disposición de quien la quiera ojear? ¿Qué se gana con ello? ¿Se pierde algo?

## Lo habitual es publicar

Desde hace tiempo, los proyectos de software libre suelen publicar (en el sentido de permitir el acceso de cualquiera) la mayor parte de la información que manejan. Esto incluye, por supuesto, el código fuente de las diferentes versiones de los programas que liberan. Pero habitualmente, hay mucho más.

Por ejemplo, normalmente está disponible no sólo el código de esas versiones *liberadas*, sino toda la historia de desarrollo: cada cambio, cada línea añadida o quitada, anotada con el momento en que se hizo, quién lo hizo... Esta información se suele mantener en un sistema de control de versiones, habitualmente CVS. Un vistazo al CVS del proyecto permite conocer, con todo detalle, su historia íntima. Se puede saber quién está haciendo más contribuciones ahora mismo, o quién las hacía hace tres años. Qué tipo de modificaciones están teniendo lugar, o qué nuevas funcionalidades se están añadiendo, y cómo. No cualquiera puede escribir en estos archivos de CVS, pero cualquiera puede usar una cuenta de sólo lectura.

También es normal que las listas de correo donde se discuten los problemas y se toman las decisiones relacionadas con el proyecto sean abiertas, en el sentido que cualquiera puede al menos leerlas, y en muchos casos apuntarse y participar en las discusiones. Quizás por este motivo hay quien tiene la idea de que en los proyectos de software libre hay muchas enemistades y muchas discusiones: porque todo tiene lugar en público, en ese escenario que supone el archivo de la lista de correo de desarrolladores. Desde luego, cada proyecto hace las cosas de una forma, pero no es raro poder asistir, en primera fila, a los debates sobre

qué funcionalidad añadir a un programa, cómo corregir un error, o si tal código es o no es de la calidad suficiente como para poder ser incluido.

Otro tipo de información muy habitual entre la que proporciona cualquier proyecto es la lista de errores conocidos (pasados y presentes) y de los arreglos (parches) que se han usado para corregirlos. Esta información suele almacenarse en los sistemas de control de errores (*bug tracking systems*), e incluye datos muy detallados sobre quién descubrió el error, en qué condiciones, qué proceso se siguió para resolverlo, si está ya resuelto o no....

Y así se puede continuar con la lista de información pública, que es diferente en cada caso, pero que habitualmente se mantiene *abierta* de forma completamente deliberada.

## La información ha de ser libre

Desde luego, no hay una única razón para que los proyectos de software libre permitan un acceso muy completo a toda la información que generan. Por un lado tenemos razones meramente prácticas, que voy a tratar un poco más adelante. Pero por otro lado tenemos también poderosas razones éticas: la información sobre un proyecto libre debería ser libre. Por ejemplo, en el Contrato Social de Debian[1] (en el que los desarrolladores del proyecto exponen su compromiso con los usuarios) se puede leer en uno de sus apartados:

> *"No esconderemos Problemas. Mantendremos nuestra base de datos de informes de errores abierta a acceso público en todo momento. Los informes que los usuarios envíen en línea se harán visibles inmediatamente al resto."*

En este caso, los desarrolladores de Debian consideran esto una obligación con sus usuarios, no sólo un asunto meramente práctico, que puede tener más o menos ventajas. Es parte de la ética que les lleva a producir su distribución de GNU/Linux, y a hacerlo de una cierta manera.

En otras palabras: en el mundo del software libre es habitual encontrarse con la opinión de que *la información ha de ser libre*. Y eso se aplica no sólo al software que se produce, sino a todo lo que le rodea. Digamos que, de alguna manera, el libre correr de la información es algo que impregna la mayor parte del mundo del software libre de una forma casi natural. Puede sorprender al principio, pero créeme, una vez que lo has probado no es algo que quieras dejar...

## Si te sientes parte, quizás participes...

Obviamente, las razones éticas no son las únicas para permitir el acceso a gran parte de la información sobre los proyectos. También hay poderosas razones

[1] http://www.debian.org/social_contract.es.html

prácticas. Y entre ellas, quizás la más clara es el efecto *implicador* que tiene discutir las cosas en público, el dejar que otros vean lo que has hecho, con todo el detalle posible.

Un día te encuentras con un problema en un programa libre que estás usando. Se te ocurre ir al sitio web que mantiene el proyecto que lo crea. Ves que se pueden enviar informes indicando errores que se hayan encontrado, y envías uno explicando ese problema. Al cabo de un tiempo, alguien te responde y te dice que el asunto se está discutiendo en la lista de desarrolladores. Te apuntas a ver qué se dice. Después de leer varias opiniones, te animas a dar la tuya. Alguien considera tu aportación y empieza a codificar un parche usando tus ideas. Tú lo ves, y le respondes refinando un poco el código. Te dice que estupendo, que lo mires en el CVS, que ya lo ha subido. Al ir a mirarlo, te das una vuelta por los fuentes y ves un sitio donde el código se podría optimizar un poquito. Como ya estás en la lista de desarrolladores, lo comentas allí... Y unas semanas después te has convertido en uno de los desarrolladores del proyecto.

Desde luego, este escenario es ideal. Muchas veces ni siquiera te animas a informar del problema que te has encontrado. Pero de vez en cuando estas cosas pasan... Si conoces a algún desarrollador de software libre, pregúntale cómo empezó. No son pocas las ocasiones en que te responderá algo como: "Encontré un problema en un programa, fui al sitio web..." ¿Te ha pasado a ti?

En cualquier caso, está claro que si se da cancha a la gente, muchas veces hay quien se anima a participar. Es más fácil conseguir colaboradores si permites acceso fácil a toda la información que necesiten, y si ayudas a que sea fácil sentirse parte del proyecto. Y es más fácil que haya quien se sienta parte del proyecto si le permites participar en los debates, y contribuir fácilmente con sus opiniones (y su código). Muchas veces la gente se pregunta eso de "¿y por qué va nadie a colaborar con un proyecto de software libre?". Sin duda, ésta es una de las respuestas posibles: porque se siente implicado en él.

## Si hay problemas, mejor que se sepan

Si se quiere mantener algo seguro, hay dos opciones: esconder los problemas que presente, o arreglarlos. Por ejemplo, si no quieres que nadie entre en tu casa, pero no tienes puerta, puedes esconder la entrada (plantando un seto delante, por ejemplo), o puedes preocuparte de poner una puerta. Y en general, te encuentras a gente que prefiere plantar el seto, y a gente que sólo se queda contento con la puerta puesta... ¿Tú de cuáles eres?

Normalmente el asunto no es tan obvio. A veces no es la puerta lo que falta, sino que hay una ventana que te has dejado abierta, y no te has dado cuenta. ¿Qué es mejor, ignorarlo, hasta que alguien trate de entrar por ella, o enterarte cuanto antes, y cerrarla? Éste es habitualmente el dilema en la seguridad de los programas de ordenador. Hay quien confía en que si aparece un error, nadie se entere, y ya se solucionará cuando se pueda. Y hay quien prefiere que el

problema sea público lo antes posible, de forma que la gente pueda colaborar en solucionarlo, y haya una presión real para que esa solución llegue cuanto antes. Es en este sentido en el que hay que entender el párrafo del Contrato Social de Debian que mencioné más arriba.

En general, en el mundo del software libre se considera que ocultar los problemas es una mala estrategia, y no sólo de seguridad. Es mejor airearlos, que participen en su resolución todos los que quieran, y que la propia presión que causa sobre los desarrolladores el compromiso de mantener la información pública les empuje a resolver los problemas lo antes posible.

El debate sobre qué hacer en caso de problemas (publicarlos o tratar de mantenerlos en secreto) es viejo, y hay muchas opiniones en un sentido y en otro. Pero una vez que has decidido dejar tu código fuente *al aire* no hay mucho donde elegir. Así que llevando al máximo la transparencia, el mundo del software libre trata de explotar las ventajas del escrutinio público. Probablemente ésta sea la razón fundamental de que un proyecto con suficiente masa crítica tenga pocos problemas de seguridad, y cuando éstos aparecen, se resuelvan de forma rápida.

## Malamente se puede ayudar si no se sabe en qué

Si ya se ha decidido colaborar con un proyecto de software libre (o simplemente, después de ir aproximándose poco a poco, un día te has encontrado metido en uno de cabeza), es mucho más fácil colaborar de forma útil si se sabe mucho sobre él. Por ejemplo, si se puede buscar de forma simple, entre la lista de errores conocidos, uno que apetezca arreglar. O si ante un problema cualquiera se pueden buscar antecedentes y debates relacionados en las listas de correo del proyecto. O si mirando un fragmento de código se puede echar un vistazo a su historia, y quizás hasta a los motivos para irlo cambiando...

En general, cuando uno comienza a colaborar en un proyecto libre, si puede ponerse pronto *al día*, y encontrar por sí mismo toda la información que necesita, será más fácil integrarse en él, y sobre todo la colaboración será más productiva. O al menos, ésa es la idea.

## La vergüenza torera

Si estás orgulloso de lo que haces, y de cómo lo haces, te gustará que todo el mundo lo vea. Por otro lado, cuando estás trabajando, normalmente no actúas de la misma forma si el fruto de tu trabajo va a ser público, y mucha gente va a tener la oportunidad de examinarlo y admirarlo (o criticarlo), que si nadie (más que quizás tu jefe) va a poder verlo. En el mundo del software propietario, sólo tus colegas más cercanos, los que están en tu propio equipo, pueden ver los resultados de tu trabajo (el código que has escrito, los errores que se han detectado en él, las decisiones de diseño que has tomado). Así que si metes la

pata, no se nota tanto... Sólo lo sabrá tu entorno más cercano, que probablemente será condescendiente (ya sabes: hoy por ti, mañana por mí).

Pero en el mundo del software libre, tus *vergüenzas* están al aire. Si escribes un código que es manifiestamente malo, o si hay en él un error de bulto, detectado desde hace meses, y no has hecho nada por corregirlo (o lo has corregido mal), cualquiera puede verlo. De alguna manera, es como trabajar en la calle, a la vista de todo el mundo. O quizás más: trabajar a la vista de muchos profesionales, como tú, que sin duda están capacitados para juzgar tu trabajo. Por eso muchas veces se dice que en el mundo del software libre es habitual la meritocracia. Es difícil que no sea así, porque aquí todo el mundo conoce la vida y milagros de los demás: basta con ojear el CVS.

En general, este control (la *vergüenza torera* de saberte expuesto a los ojos de tus pares) ha mostrado ya ser muy útil para conseguir la excelencia en otros ámbitos (como por ejemplo, el trabajo científico), y probablemente está en la raíz de las altas productividades, y de la buena calidad del trabajo de tantos desarrolladores, que se observa en el mundo del software libre.

## ¿No será esto de aplicación en muchos otros ámbitos?

Desde luego, mucho de lo que se ha comentado hasta aquí no es exclusivo del software libre. Es más que posible que otros ámbitos, y sobre todo los relacionados con el proceso de información y con la creación de conocimiento, se beneficiarían mucho de poner gran cantidad de información *interna* a disposición del público en general. Pero hay que reconocer que esto supone un gran cambio en la mentalidad habitual, centrada en *lavar los trapos sucios en casa*, y en mostrar hacia el exterior sólo lo que es percibido como buena imagen.

Eso sí, el mundo del software libre aporta un caso de ejemplo de cómo tener *paredes de cristal* puede ser beneficioso para el desarrollo de un proyecto. Pero aún queda por saber si realmente hay relaciones causa-efecto (por ejemplo: los proyectos libres que son más transparentes, ¿son también los más productivos, y los de mayor calidad?), y si esta forma de funcionamiento es realmente trasladable a otros ámbitos.

Una vez más, el mundo del software libre está transitando por terreno inexplorado... pero muy prometedor.

# KDE o GNOME, ¿es ésa la cuestión?, ¿es la cuestión GNOME o KDE?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 6, pág. 12-13, Febrero de 2001

Hace unos años no disponíamos de ningún entorno de escritorio en el mundo del software libre. Este vacío lo han llenado dos proyectos que hoy son muy conocidos: GNOME y KDE. Cada uno proporciona un sistema de desarrollo propio y dispone de su buena cantidad de aplicaciones. Ambos funcionan en muchas plataformas, y ambos están compuestos completamente por software libre. Ambos tienen detrás organizaciones y empresas que los apoyan, y nutridas cantidades de usuarios satisfechos. Todo parece mucho mejor que hace unos años...

Pero sin embargo, hay dos polémicas recurrentes sobre esta situación. Una consiste en preguntarse si KDE o GNOME, si GNOME o KDE. Cuál es mejor, cuál es más libre, cuál es más rápido, más usable, cuál es mejor para tal o cual tarea. La otra trata sobre la duplicación de esfuerzos: ¿No sería mejor tener un solo entorno de escritorio libre, y centrar en él los esfuerzos de los desarrolladores de software libre? Voy a echar más leña al fuego terciando en ambas.

## ¿Cuál lava más limpio?

Igual que hay gente del Madrid o del Barcelona, hay gente de KDE o de GNOME. No son mayoría entre los usuarios de ninguno de los dos sistemas, pero son muy ruidosos. Igual que hay quien apoya a su equipo *manque pierda*, hay quien apoya a su entorno por motivos relativamente irracionales. En algunos casos porque es el único que han probado. Otras veces porque alguien les ha dicho que es mejor. O porque una vez probaron el otro y se les colgó. O porque les gusta mucho cómo habla Miguel de Icaza, o porque les parece un chulo. O por alguna entre miles de razones más. A veces, incluso por motivos más racionales como pruebas de rendimiento, funcionalidad de algunas aplicaciones, u otros argumentos más o menos objetivos. Hasta aquí todo normal. A mí hace tiempo que me gusta más XEmacs que Emacs, Debian que Red Hat, bash que tcsh,... Cada uno tenemos nuestras preferencias, que para eso somos cada uno.

Pero en el caso de GNOME y KDE la discusión es muchas veces exagerada. No hay forma de sacar el tema en BarraPunto sin que la cuenta de comentarios suba hasta las estrellas. Aparecen por todas partes fanáticos de los dos sistemas. Los argumentos llegan a bordear los insultos. Desde luego, hay pocos temas que levanten tantas pasiones en el mundo del software libre, salvo quizás las preferencias por la licencia GPL o por la BSD (lo que me da una idea para otro comentario, para otro día). Se discuten los modelos de desarrollo, la personalidad

de los líderes de los proyectos, la velocidad de ejecución, el espacio que ocupa en disco y el color de la alfombrilla de ratón que mejor hace juego con el tema por defecto de cada uno de ellos.

Uno acaba un poco cansado de todo esto. ¿Es que no es posible decidir de una vez por todas cuál es el mejor de los dos sistemas, y zanjar la discusión? ¿Es que no hay estudios más o menos imparciales donde se analicen todos los aspectos, y se llegue a alguna conclusión? La respuesta es no. No sé si afortunada o desgraciadamente, pero es no. Y es difícil pensar que podría ser de otra forma. Los dos sistemas son muy complejos, y simplemente hay demasiados aspectos a tener en cuenta como para poder decir de forma binaria, marcando a uno como el ganador y arrojando al otro al abismo.

Pero podemos ir un paso más allá: ¿Realmente importa tanto esta discusión? Después de empezar asistiendo como espectador a estas batallas verbales, después de picar más de una vez y tomar partido en ellas, después de aburrirse de ellas, algún día uno llega a la pregunta del millón: ¿Realmente importa tanto?

Si no fuera porque hay tantas pasiones por medio, la respuesta parece fácil. Si estás contento, sigue usando lo que prefieras. Y no te metas con el resto. Fácil de decir, desde luego. Pero ¿y si resulta que *el otro* es mejor? Y vuelta a empezar. Pero no, hombre, da igual. No le des vueltas. Recuerda: ¿Realmente importa tanto? Y cuando uno está finalmente a gusto con su GNOME, vienen los de KDE y sacan nueva versión. Y vuelta a empezar... ¿Verdad que es cansado?

## Dos mejor que uno o uno mejor que dos

Ahora más en serio. ¿Es bueno tener dos entornos de escritorio con metas similares? ¿No es una duplicidad de esfuerzos absolutamente innecesaria? ¿Es que estamos tan sobrados en el mundo del software libre que podemos permitirnos estos lujos? ¿Para cuándo la gran unificación entre KDE y GNOME?

Una de las opiniones que más se oyen en relación a este tema es que *no es bueno duplicar esfuerzos*. Otra versión de lo mismo es *mejor centrarse en hacer una buena aplicación que dos mediocres*. Sin embargo, yo creo que para los gustos hay colores, que no es bueno meter todos los huevos en el mismo cesto, y que más vale pájaro en mano que ciento volando.

### Para los gustos hay colores

Tanto GNOME como KDE tienen sus propias metas, su propia forma de organizarse, su equipo de personas. Cada uno funciona de forma diferente, y los resultados de su trabajo son también diferentes. Claramente, si tenemos en cuenta que ambos se mueven en el mismo ámbito, y ambos están dentro de la misma comunidad (la del software libre), esta situación produce inevitablemente duplicidad de esfuerzos y trabajos repetidos. Y sin embargo, a todo el mundo nos gusta elegir. Hay quien prefiere coches rápidos, quien los prefiere grandes, y quien prefiere ir en bicicleta o andando. ¿Por qué no vamos a querer poder

elegir también dentro del mundo del software libre? Mientras KDE y GNOME se mantengan libres, podemos decidir cuál de los dos preferimos de acuerdo con nuestros gustos y según nuestras necesidades. O utilizar lo que nos parezca mejor de cada uno de ellos.

Además de permitir a los usuarios que elijan, hay otra consecuencia deseable: los dos proyectos compiten por la aceptación de los usuarios. Ninguno de ellos puede quedarse muy atrás, o se arriesgará a perder la masa crítica que necesita para desarrollarse. La comunidad del software libre no es tonta: si uno de los dos sistemas llega a ser claramente mejor que el otro la mayoría del desarrollo, de los recursos, y de la base de usuarios se irá con él. Y esto impone una formidable presión en ambos proyectos, que les obliga a mejorarse cada día. De lo cual no podemos más que beneficiarnos todos.

De todas formas, y teniendo en cuenta la competencia feroz entre KDE y GNOME, no hay que olvidar que ambos están desarrollándose dentro de la misma comunidad, y que el uso de licencias libres garantiza que cada uno de ellos puede aprovecharse de mucho de lo que desarrolla el otro. Al fin y al cabo, es bueno recordar que estamos en una de las pocas comunidades donde copiar (dando el crédito adecuado, claro) está bien visto. De esta manera estos dos sistemas no sólo compiten, sino que además colaboran, incluso sin plantearse esta colaboración como objetivo, porque cada uno pone a disposición de la comunidad (y en ella está su proyecto *rival*) todo el código fuente que ha desarrollado.

## No es bueno meter todos los huevos en el mismo cesto

Es bien conocido que si se depende de algo de forma crítica, es mejor duplicarlo. Si falla una de las instancias, siempre quedará la otra. De la misma forma, creo que es muy bueno tener dos grupos de desarrolladores trabajando en el mismo tipo de cosas. Como cada uno tiene una organización y una forma de hacer las cosas diferente, es poco probable que ambos fallen y no consigan sus objetivos. Y no sólo en términos absolutos. También en términos relativos, cada uno de nosotros tenemos unas necesidades que potencialmente cualquiera de los dos proyectos puede satisfacer. Si, por los motivos que sea, pasa el tiempo y uno de los dos proyectos no lo consigue, siempre podemos volver nuestra vista al otro: quizás tengamos allí lo que necesitamos.

## Más vale pájaro en mano que ciento volando

Por encima de todo, la discusión sobre si se duplican o no esfuerzos, y sobre si es bueno o no tener dos proyectos en lo mismo es bastante estéril. Ha quedado claro muchas veces que vamos a tener GNOME y KDE para rato, con más o menos coordinación. Y cada uno de ellos lleva ya tiempo produciendo resultados en forma de aplicaciones utilizables. Por lo tanto, es mucho mejor aceptar las cosas como son que perder el tiempo quejándose y explicando cómo podrían ser. Si por algún motivo alguien cree que es conveniente una unificación de los dos

proyectos, que trabaje para conseguirlo. Pero creo que sería bueno que no gaste su tiempo en desprestigiar a uno de ellos (o a los dos). Que no se centre demasiado en lo que podría ser, sino que trabaje en lo que es. Y sobre todo, creo que es especialmente bueno no escarbar en los problemas personales que pueda haber entre desarrolladores de uno y otro proyecto (o incluso inventar esos problemas): no lleva a ningún sitio. O al menos a ningún sitio que beneficie a alguien. Como dice un amigo, mejor discutir menos y tirar más líneas (de código).

## Conclusiones

En el mundo del software libre tenemos la suerte de tener dos proyectos que nos están trayendo entornos de escritorio estupendos. Tenemos la suerte de que ambos están mejorando cada día, produciendo más y más aplicaciones que nos ayudan a trabajar más a gusto. Tenemos la suerte de poder elegir, y de poder combinar lo mejor de ambos mundos. ¿Realmente crees que sería mejor la situación si sólo hubiese KDE o GNOME? ¿No es mucho mejor tener KDE **y** GNOME? ¿Y no es estupendo que ambos sean software libre?

Resumiendo, creo que la pregunta *¿GNOME o KDE?*, que podría tener sentido desde un punto de vista estrictamente técnico en un momento dado, no tiene mucho sentido desde un punto de vista más global. Igual que no es mejor el café con leche o el café solo, a unos nos gustará más GNOME y a otros KDE. Y a otros muchos ambos. Y si no nos gustan, siempre podemos colaborar en los proyectos, y mejorarlos (aunque probablemente colaborar en uno sólo ya sea suficiente).

Y luego están, claro, los que prefieren un terminal en modo texto, y pasan completamente de GNOME, KDE e incluso del ratón y los iconos... Pero eso sí que es otra historia.

# La educación y el conocimiento libre

Uno de los denominadores comunes de muchos de los autores cuyos artículos se recogen en esta colección es su estrecha relación con el mundo de la docencia, en particular con la docencia universitaria en el campo de la informática. Por eso, cuentan con una especial *sensibilidad* para temas relacionados con la educación y la difusión de conocimiento.

El uso de herramientas informáticas libres se está convirtiendo en habitual en institutos y universidades en los últimos años. Esto es debido a varias características del software libre que se tornan en grandes ventajas de cara a la docencia, como se puede leer en *Software libre en la enseñanza informática*, el primero de los ensayos de esta parte. Piénsese, por ejemplo, en el hecho de que los profesores pueden distribuir los programas de forma gratuita a sus alumnos o, muy importante en el caso de estudios informáticos, se puede inspeccionar el código fuente.

La posibilidad de inspeccionar el código fuente y, en general la estructura de los programas, puede parecer un hecho menor, pero sin lugar a dudas influye de manera notable en los planes de estudio de las carreras informáticas. Así, de las aplicaciones que casi todo usuario suele utilizar -sistemas operativos, suites de ofimática y recientemente aplicaciones para Internet-, sólo los sistemas operativos y las aplicaciones para Internet suelen ser estudiadas de manera minuciosa en la universidad; las asignaturas de creación de herramientas ofimáticas brillan por su ausencia. Entre otras razones, podemos suponer que esto es debido a que hasta hace poco no hemos estado en disposición de conocer cómo se construyen ni de estudiar de cerca casos prácticos de esas herramientas. Ciertamente, esto está cambiando gracias al software libre y a las nuevas posibilidades que ofrece.

En este sentido, las universidades punteras están empezando a cambiar de rumbo y empiezan a florecer iniciativas para dar difusión a sus cursos, algo que hasta ahora lo hacían como mucho los profesores a nivel particular. Esto no es otra cosa que una generalización de las ideas que hacen al software *libre* y aplicadas al conocimiento. En *De cómo el conocimiento puede ser libre* se presenta una iniciativa -probablemente la más conocida- por parte de una prestigiosa universidad americana. Si ya ahora es una referencia, a partir de ahora y con iniciativas como ésta, lo será más.

Finalmente, en *¿Qué tiene que estudiar un informático?* se plantean una serie de ideas que ya han sido expuestas en los dos artículos anteriores, pero esta vez orientadas de manera concreta a las carreras de ingeniería en informática. Los

lenguajes de programación, los intereses de la industria informática, la deontología profesional, etc. serán algunos de los temas tratados.

# Software libre en la enseñanza informática

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

Hace ya tiempo que, en mayor o menor medida, los ordenadores han entrado en las escuelas, en los institutos, y desde luego en las universidades. En muchos casos con ellos se practican conocimientos específicamente informáticos, pero cada vez más se usan sobre todo como herramienta para enseñar otro tipo de disciplinas, o simplemente para permitir a los alumnos practicar con herramientas genéricas (ofimáticas, de consulta de web, etc.).

La inmensa mayoría de estos ordenadores utilizan software propietario, y en particular alguna versión de Microsoft Windows y Microsoft Office. Sin embargo, la elección de estos programas raramente es una decisión meditada, ni suele estar basada en un análisis de las opciones disponibles. Es más, en muchos casos ni siquiera los responsables de esta decisión son consientes de que existen otras opciones. Pero estas otras opciones existen, y entre ellas destaca por sus ventajas la basada en software libre. ¿Es ya hora de que el software libre ocupe en el mundo de la educación un lugar destacado?

## La situación actual

La educación (reglada o no) relacionada con la informática es hoy día un monocultivo de algunas marcas de software propietario. Sin realizar en muchos casos ningún estudio previo, se elige como plataforma para la formación en técnicas relacionadas con la la informática la que se percibe como *la más habitual*. Sin pararse a pensar si esta es la mejor opción posible, se llega a confundir la introducción a la informática con un curso de introducción a cierto sistema operativo, los conocimientos sobre ofimática con el conocimiento de una cierta marca de programa ofimático, o incluso la navegación por el web con el manejo de cierto programa navegador. En general, mucha gente se ha aproximado al ordenador en un entorno donde la suposición implícita es que saber de informática es lo mismo que saber manejar ciertas herramientas propietarias, y fundamentalmente Microsoft Windows y Microsoft Office.

En los casos en los que esta decisión se ha tomado mediante algún tipo de proceso racional, los motivos que suelen aducirse son los siguientes:

- Es mejor enseñar el uso de la plataforma dominante en el mercado, porque así lo enseñado será más útil al alumno.
- Los propios alumnos piden que se les enseñe el uso de ciertos programas, y piensan que si se usan otros, los conocimientos les van a ser de menos utilidad.
- No hay muchas alternativas, y en cualquier caso, no hay alternativas con ventajas claras sobre el uso de la plataforma dominante.

¿Son estas razones válidas? ¿Merece la pena estudiar si es posible usar otro tipo de programas para estas tareas? Mi planteamiento es que sí. Y eso implica la negación de las razones anteriores. Creo que no es mejor enseñar el uso de ninguna plataforma en particular, que sí hay alternativas, y que los alumnos pueden pedir lo que sea, pero la labor del docente es precisamente orientarles sobre este particular como parte de la formación informática que les debe impartir.

## Otro planteamiento para la selección de plataforma informática

Cuando se enseña carpintería no se enseña cómo usar una marca determinada de martillos o de sierras eléctricas. Cuando se enseña a escribir no se enseña el uso de una marca de plumas o bolígrafos determinada. ¿Por qué cuando se enseña informática, sí parece razonable enseñar a usar una determinada *marca* de programas? ¿Hay razones para eso, o simplemente hemos perdido nuestro sentido común?

Yo creo que ocurre más bien lo segundo. No veo razones objetivas para que cuando se enseña informática, y especialmente cuando se enseña la informática como herramienta, deba hacerse algo distinto de lo que se hace en otros contextos. Por ejemplo, creo que debe enseñarse cómo funciona un procesador de texto en general, y no los detalles del uso de Microsoft Word (o de ningún otro procesador de texto) en particular. Naturalmente habrá que hacer unas prácticas, y en ellas habrá que utilizar una herramienta dada. Pero en una clase de carpintería no se atenderá en las clases prácticas a los detalles de las herramientas de cierta marca, sino que se utilizarán de la forma lo más genérica posible. De la misma forma, en la enseñanza de informática deberían utilizarse las herramientas de la forma lo más genérica y reutilizable posible. Así, podría usarse Microsoft Word para mostrar los aspectos genéricos de un procesador de texto, y para fijar las ideas que se hayan introducido en las clases teóricas (si es que hay clases teóricas).

Si las cosas se hacen de esta forma, ya no tiene mucho sentido tratar de usar la herramienta que más usuarios tiene. Lo más razonable será usar la herramienta que más ventajas docentes presente. Si la enseñanza se hace de forma adecuada, y el alumno aprende realmente el uso genérico de un tipo de herramientas, le será fácil y rápido adaptarse a un programa dado de esa categoría.

## Las ventajas del software libre en la educación

Si estamos de acuerdo en este planteamiento docente, podemos pasar a ver cuáles son las ventajas docentes que presenta el software libre para la enseñanza de la informática. Entre otras, creo que las siguientes son las más importantes:

- El software libre puede adaptarse a las necesidades docentes de un curso dado. Puede, por ejemplo, modificarse para ofrecer a los alumnos una versión simplificada. O darle una apariencia adecuada a los conocimientos del

alumno (por ejemplo, similar a la de las herramientas con las que el alumno está familiarizado).

- Si se usan programas libres, el alumno puede reproducir todo el entorno de prácticas, con total exactitud, en cualquier otro ordenador. En particular, por ejemplo, en el ordenador de su casa, donde podrá practicar cómodamente. Y todo esto, naturalmente, sin ningún problema de licencias, y sin costes extra para el alumno. Así, para cada curso se podría estampar un CD que incluya todas las herramientas utilizadas, que se repartiría a los alumnos para que saquen sus propias copias.
- Además de las herramientas básicas utilizadas en el curso, es fácil y económico utilizar marginalmente otras similares, para que el alumno pueda experimentar con las diferencias entre herramientas parecidas. Por ejemplo, en un curso donde se enseñe a navegar por Internet, puede usarse Mozilla como herramienta básica, pero también poner a disposición de los alumnos Konqueror y lynx, para que puedan jugar también con ellos. De hecho, los alumnos interesados podrán utilizar una gran cantidad de programas, que se pueden incluir en el CD del curso, como complemento a las enseñanzas básicas.
- En el caso de que la enseñanza sea para informáticos, para gente que puede entender (y tiene que entender) las interioridades de las herramientas, la disposición del código fuente es fundamental. Esto permite, con gran facilidad y sin problemas de licencias ni acuerdos especiales con los fabricantes, ver cómo están hechas algunas herramientas reales, de calidad comercial. Y de esta forma, enseñar con el ejemplo, que es una de las mejores formas de enseñar informática.
- Si todo el software utilizado es libre, el docente puede ponerlo a disposición de otros docentes. De esta forma se pueden preparar paquetes, disponibles mediante Internet, que incluyan la documentación y los programas usados. Así, el mismo curso podrá ser reproducido en cualquier otra parte del mundo.
- En general, parece razonable que las entidades educativas, y muy especialmente las que se financian con dinero público, no favorezcan unas empresas sobre otras. De hecho, el favorecer a una empresa sobre otra en la educación es especialmente grave, pues da a la empresa favorecida una ventaja enorme sobre la competencia: los alumnos están formados para utilizar sus productos, y por tanto preferirán usarlos frente a los de la competencia, incluso si son peores o más caros. Con el software libre esto no ocurre, ya que cualquier empresa puede comercializar y vender servicios para cualquier producto libre. Por ejemplo, aunque hoy es SUN quien mantiene y comercializa OpenOffice.org, no hay motivos para que cualquier competidor suyo no pueda hacer lo mismo.

Como puede verse, estas ventajas del uso de software libre en la enseñanza no lo son sólo frente a un programa propietario dado, sino frente a cualquier programa propietario. Como ya se ha explicado, simplemente por el cambio de Microsoft Office por OpenOffice.org, por ejemplo, no experimentaremos en toda

su amplitud estas ventajas. Es preciso cambiar también el enfoque de la enseñanza, pasando de mostrar los detalles de un programa dado a explicar los fundamentos generales de un tipo de programas.

## ¿Está GNU/Linux suficientemente maduro?

Pero aún suponiendo que estemos de acuerdo en que el software libre tiene ventajas en el entorno educativo, es preciso que sea posible enseñar con él. En otras palabras, ¿hay software libre con calidad y estabilidad suficiente para poder enseñar usándolo? Y más concretamente, quedándonos en el mundo GNU/Linux, ¿está GNU/Linux suficientemente maduro como para ser una opción a la hora de elegir plataforma?

Naturalmente, la respuesta a esta pregunta depende mucho del tipo de enseñanzas al que nos estemos refiriendo. Desde hace años, es común utilizar entornos GNU/Linux para cursos de programación, sistemas operativos o redes de ordenadores en universidades de todo el mundo. Luego en esos ámbitos, la respuesta no puede ser más que un simple *sí*. Pero... ¿qué ocurre cuando estamos hablando de clases de introducción a la informática, o de ofimática, o en general de clases para alumnos con pocos conocimientos informáticos? En otras palabras, ¿está GNU/Linux listo para su uso en cursos donde se enseña informática sólo como una herramienta?

Creo que hace unos pocos años, la respuesta a esta pregunta era un *no* rotundo, o como mucho, un tímido *a veces*. Sin embargo, hoy día estamos ya muy cerca, o hemos llegado, al *sí* rotundo. La instalación de GNU/Linux es cada vez más sencilla. Los sistemas de instalación de las distribuciones actuales compiten en sencillez con cualquier otro sistema propietario, con lo que los alumnos pueden instalarse GNU/Linux en casa para practicar. Los entornos como GNOME y KDE hacen fácil el uso del sistema para los usuarios *novatos*: ya no hace falta conocer las órdenes más habituales de Unix para manejar una caja GNU/Linux. Y por fin tenemos las aplicaciones que permiten trabajar en muchos ámbitos. Por ejemplo, pueden mencionarse dos donde las cosas han cambiado claramente en los últimos años: el tratamiento de imágenes y la ofimática. En el primero, programas como el GIMP permiten la enseñanza de prácticamente cualquier concepto relevante. En el segundo, la disposición de juegos de aplicaciones como KOffice u OpenOffice.org permiten que ya se pueda aprender a manejar procesadores de texto u hojas de cálculo usando sólo software libre.

Hay que reconocer que aún hay pocas experiencias en este campo. Pero según las herramientas mencionadas se van haciendo más conocidas, y los docentes aprenden las ventajas que tiene su uso, iremos viendo cómo más y más cursos las utilizan para su parte práctica. De hecho, tengo la impresión de que el único obstáculo importante que tendrá el uso de software libre en la educación informática dentro de muy poco tiempo será el rechazo por parte de alumnos poco informados a no tener prácticas con las herramientas *líderes*, y la falsa per-

cepción de que están recibiendo una enseñanza de segunda categoría porque los programas que usen puedan descargarlos, gratuitamente, de la red.

## Para terminar...

Creo que el uso del software libre en la educación informática tiene muchas ventajas. Pero lo más importante no es simplemente cambiar en la docencia práctica un programa propietario por otro libre, sino cambiar el enfoque de la enseñanza. En lugar de enseñar los detalles del funcionamiento de un programa concreto, enseñar los fundamentos de un tipo de aplicaciones, qué tipo de cosas pueden hacerse con ellas, y cómo realizar tareas típicas utilizándolas. Si hacemos este cambio de planteamiento, que es de por sí muy deseable, el paso al uso de programas libres será más fácil, y permitirá un proceso educativo mucho más productivo.

Y en cualquier caso, si no estoy equivocado, ya hemos llegado al punto donde el entorno GNU/Linux sirve para enseñar al menos tan bien como cualquier otro. Ahora sólo hacen falta docentes que se atrevan a dar el paso que supone salirse del camino tradicional y entrar en una nueva vía. Docentes que sean capaces de repensar sus cursos, y el planteamiento de sus prácticas. Docentes que quieran ser la vanguardia de la enseñanza de informática... Y alumnos que sean capaces de aprovechar todas estas novedades.

# De cómo el conocimiento puede ser libre

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Agosto de 2002

En abril de 2001 el MIT (Instituto Tecnológico de Massachusetts) lanzó el proyecto OpenCourseWare (OCW, algo así como *material para cursos abierto*). Este proyecto consiste en la publicación, libremente y en la Red, del material usado en la mayoría de sus cursos, de forma organizada, uniforme, y con acceso fácil. Este proyecto supone un aire fresco en el panorama de la educación superior, dominada por los intentos de controlar la información (tanto la producida para la docencia como la que es resultado de la investigación). Contra las tendencias a la privatización del conocimiento, la iniciativa del MIT trata de promover la máxima difusión del conocimiento.

Si esta iniciativa la hubiera propuesto una universidad desconocida, su impacto sería discutible. Pero habiendo partido de una entidad del prestigio del MIT su efectividad, su solvencia y su influencia sobre el desarrollo del modelo de universidad en el siglo XXI pueden ser determinantes.

## Los ideales de la Ilustración

La lucha por la libertad del conocimiento está en el origen del progreso de la humanidad. Una de las revoluciones sociales más destacadas, la Ilustración, supuso en gran medida precisamente esto: hacer del conocimiento un bien público, al que todos los ciudadanos pudieran tener acceso. Con el uso de las tecnologías disponibles en la época (fundamentalmente la impresión sobre papel) se trató de acercar el Conocimiento, con mayúscula, a grandes capas de la sociedad. Los efectos fueron parciales, naturalmente. Gran parte de la sociedad no podía leer. La calidad de las cadenas de distribución de obra impresa dependían mucho de la geografía. Y no todo el mundo tenía suficientes recursos para adquirir todas las obras impresas en que podía estar interesado.

Con el tiempo, grandes esfuerzos de instrucción pública redujeron el analfabetismo. Los nuevos medios de transporte de los siglos XIX y XX hicieron posible poner una obra impresa en casi cualquier parte del mundo. Y la red de bibliotecas públicas aseguró, al menos en cierta medida, el acceso de todos al conocimiento, a la cultura. A finales del siglo XX, muchos promotores de la Ilustración habrían pensado que uno de sus fines estaba en vías de conseguirse.

Sin embargo, la aparición de Internet, y con ella de la publicación electrónica accesible desde cualquier parte del mundo a coste prácticamente nulo ha cambiado las reglas del juego. Por un lado, ha hecho posible la difusión universal de información, sin necesidad de transporte físico ni de impresión en papel. Basta con poner una obra en el web para que cualquiera pueda acceder a ella. Pero por

otro lado, ha hecho posible un nuevo negocio, la venta del conocimiento de forma completamente racionada y controlada. Con los nuevos métodos técnicos y las legislaciones de limitación de acceso a la información, el control de quién tiene acceso al conocimiento queda férreamente en manos de los editores de contenidos. Y la tendencia parece ir en la dirección de darles más y más poder en detrimento de lo que pueden hacer los *consumidores de información*. Además, ante la posibilidad de comercializar la información para mercados enormes (potencialmente todo el mundo), los incentivos para no publicarla libremente (incluso si a medio plazo no hay planes de comercialización) anima a mantener los contenidos tan controlados y secretos como sea posible.

## La idea del MIT

Precisamente preocupados por esta *privatización del conocimiento* en su ámbito (la enseñanza universitaria), el MIT ha anunciado su proyecto OpenCourseWare. Ya el nombre del programa nos da pistas sobre su inspiración: el software libre (u Open Source software). La idea es, en parte, trasladar el modelo de software libre (o al menos parte del modelo) a la producción de materiales para cursos académicos.

El proyecto fue anunciado en abril de 2001 como un compromiso para dejar en el web, libres para su uso no comercial, los materiales de casi todos los cursos impartidos en el MIT, unos dos mil. Al cabo de año y medio, el proyecto da sus primeros frutos, con la publicación del material de unos veinte cursos, a modo de ejemplo y de prueba del proyecto. Estos materiales quedarán, por lo tanto, a disposición de cualquier estudiante, profesor, o en general, cualquier persona interesada en consultarlos. Además, se pretende ofrecerlos de una forma integrada, de fácil uso, y con énfasis en la relación entre unos materiales y otros.

Entre otras preocupaciones de los profesores del MIT que han llevado a la concepción de este proyecto destaca la posibilidad, cada vez mayor, de que los contenidos de sus cursos sean distribuidos sólo de forma restringida, y no sirvan para el fin último de difundir y ayudar a ampliar el conocimiento. Citando la nota aparecida en MIT News (traducción libre):

> "[El programa OpenCourseWare] expresa nuestra creencia en la forma en que la educación puede avanzar: mediante la constante ampliación del acceso a la información, y mediante la invitación a otros a participar."

Por lo tanto, OCW marca una diferencia sustancial frente a la tendencia que se puede observar cada vez más en las universidades de todo el mundo. En lugar de pretender controlar lo más posible los materiales producidos, buscando rentabilizarlos económicamente, propone justamente lo contrario: permitir el acceso libre, para que su impacto sea lo mayor posible.

## ¿Dónde está lo novedoso?

Si el proyecto OCW se hubiese limitado a la publicación del material de cursos ya sería extraordinario, pero no tan novedoso. Porque aunque es bastante extraño que una universidad (y menos una de tanto prestigio) anime institucionalmente a la publicación libre de estos materiales, es relativamente común que muchos profesores, a título casi personal, lleven años haciéndolo. Una rápida consulta en cualquier buscador de Internet nos dará cientos y miles de referencias a materiales de cursos que están disponibles en Internet para quien quiera consultarlos.

¿Dónde está lo nuevo entonces? Por un lado, la decisión institucional, después de un proceso de reflexión, y con el apoyo mayoritario del claustro de profesores y de la dirección de la universidad ya es, como he comentado, algo excepcional. Pero el MIT ha ido un paso más allá. No sólo va a permitir que sus profesores *cuelguen* su material en la Red. Lo va a fomentar. Va a proporcionar todo este material de forma integrada, relacionada y accesible. Y además, la cantidad del material (y su calidad) va a ser también extraordinaria: dos mil cursos en todas las ramas de la técnica. Y va a dedicar considerables recursos a hacer todo esto realidad.

A estas alturas es muy pronto para estimar con certeza el impacto que tendrá esta iniciativa. Pero es posible que cambie en gran medida los usos habituales en cuanto a material para cursos se refiere. Por un lado cada libro, cada juego de apuntes, cada manual, se va a comparar (por los alumnos, por los profesores) con el que estará disponible en el MIT para esa materia. Rápidamente cualquiera que los compare se va a formar una idea sobre su calidad, al menos en relación al material elaborado en el MIT. Y probablemente el MIT es un buen competidor en este negocio... Va a ser difícil que un profesor trate de imponer, por ejemplo, sus apuntes a los estudiantes, si estos apuntes son mediocres.

Por otro lado, es de suponer que la influencia del MIT en las enseñanzas técnicas de todo el mundo (que ya es ciertamente apreciable), aumentará notablemente. Es de esperar que muchos profesores en todo el mundo elijan como textos base los que proporcionará el MIT, u otros basados en ellos. Naturalmente, el retorno que recibirá el MIT en términos de imagen no será despreciable.

Desde otro punto de vista, el impacto en los países menos desarrollados, donde el acceso a materiales docentes de buena calidad es más difícil por razones de coste, será también grande. En cuanto el acceso a Internet se vaya haciendo habitual (y hasta cierto punto ya lo es, al menos en las instituciones universitarias de estos países) no habrá ninguna barrera económica para el acceso a textos que estarán sin duda entre los mejores en su campo.

En cuanto a la colección de material que proporcionará OpenCourseWare en sí, la disposición de esa cantidad de obras, bien estructuradas, interrelacionadas y fácilmente accesibles se va a convertir en una gran enciclopedia de obligada consulta en todos los campos del conocimiento que va a cubrir. Muchas veces se ha dicho que Internet es la Enciclopedia de nuestra época. Proyectos como este probablemente van a dotar de significado literal a esta frase.

Por último, es de esperar un cierto efecto arrastre. Va a ser muy difícil competir con materiales *con licencia tradicional* frente a estos. Quizás eso (junto con las ventajas ya mencionadas) anime a otras instituciones a tomar decisiones similares.

## La realización del proyecto

El anuncio del MIT no ha sido sólo una declaración de intenciones. Inmediatamente después de su presentación pública, se aseguró financiación para su lanzamiento: unos 11 millones de euros, para los dos primeros años (dinero aportado por dos fundaciones privadas). Durante la duración del proyecto (10 años) se espera que el MIT tenga que desembolsar alrededor de 10 millones de euros anuales, que procederán de sus propios fondos y de donaciones.

Estos considerables recursos se han utilizado para establecer un grupo estable que se encargue del diseño detallado y de la coordinación. Además, se va a usar el servicio de publicaciones del MIT, y varios de sus centros van a colaborar específicamente en las fases de elaboración y publicación de material. Por último, naturalmente, los profesores del MIT que quieran (la participación del profesorado será voluntaria, pero se espera que sea masiva).

El proyecto ya está empezando a dar sus frutos, con la publicación de los primeros materiales. Su impacto en la comunidad universitaria mundial está empezando a percibirse, en la forma de discusiones y debates sobre cómo se debe gestionar la producción intelectual. El tiempo dirá si este impacto es duradero.

## Para terminar...

El proyecto OpenCourseWare es, sin duda, una iniciativa excepcional. Todavía es pronto para saber si supondrá un punto de inflexión en la historia de la enseñanza, o si el tiempo la convertirá en una mera anécdota. Pero desde luego es una forma novedosa de aproximarse a las nuevas posibilidades y a los nuevos problemas que nos ofrecen los nuevos mecanismos de transmisión de información. Más allá de tratar de reproducir modelos antiguos en el mundo de las comunicaciones digitales, OCW supone un intento de repensar, desde sus propios fundamentos, lo que se puede hacer cuando la tecnología base ha cambiado.

Porque toda la comunidad docente está segura de que Internet y su uso universal va a cambiar muchos esquemas en la enseñanza. Pero hay pocas iniciativas que propongan caminos nuevos, y nuevos esquemas que aprovechen y se adapten a esta situación. Quizás, frente a las opiniones más pesimistas que nos muestran un futuro con el conocimiento parcelado, con fuertes restricciones de acceso a la información, y con trabas para la difusión universal de la cultura, OCW nos ofrece una visión mucho más optimista. En ella, se muestra cómo pueden aprovecharse las nuevas oportunidades para conseguir una difusión del conocimiento

sin precedentes en la historia de la humanidad. Desde luego, las posibilidades que nos ofrece esta visión, en caso de realizarse, son difíciles de imaginar...

**Nota:** El proyecto OpenCourseWare está en `http://web.mit.edu/ocw`

# ¿Qué tiene que estudiar un informático?

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

¿Qué tienen que estudiar los ingenieros informáticos? Es curioso que una profesión tan extendida con titulaciones en casi todas las universidades españolas tenga tan poco claro lo que tienen que estudiar sus ingenieros.

## Introducción

Mi artículo de este mes no trata de los temas que habitualmente ocupan el espacio que me conceden de vez en cuando en esta revista. No voy a tratar los problemas de los derechos de autor, ni los derechos de los internautas, etc. Quiero entrar en otro de los aspectos que me preocupan y que no es otro que la formación y, dentro de ella, el área que más conozco, la formación de los informáticos.

Una vez escrito este artículo, he de completar esta introducción para indicar que en el fondo sí trato los temas de siempre, fundamentalmente porque la formación de los profesionales informáticos está muy relacionada con el estado de la informática y especialmente con su futuro.

## ¿Qué se enseña en informática?

Los estudios de informática entraron en la universidad en España en 1977 con la creación de la Licenciatura en Informática, que en los años 90 se convirtió en Ingeniería Informática.

Sus primeros planes de estudio estaban muy influenciados por la procedencia de la mayoría de su profesorado. En algunas Facultades los profesores provenían fundamentalmente de áreas de ingeniería -ingeniería de telecomunicaciones e ingeniería industrial fundamentalmente- y en otras facultades de áreas de ciencias -matemáticas y física fundamentalmente-. Los primeros títulos oficiales "Licenciados / Diplomados en informática" parece que se decantaban más por esta segunda corriente, mientras que los actuales "Ingenieros / Ingenieros Técnicos en Informática" parece que se acercan más a la primera.

Esta diferencia de culturas llevó al choque entre dos concepciones distintas sobre los planes de estudio en informática. Los primeros con mayor énfasis en asignaturas aplicadas como sistemas operativos, redes de ordenadores, ingeniería del software, etc. Los segundos con más peso en el estudio básico, con mayoría de asignaturas como cálculo, álgebra, matemática discreta, teoría de autómatas, etc. El resultado final de los planes de estudio obviamente conjuga ambas visiones por imperativo legal (la troncalidad fijada en el BOE), aunque la relación de fuerza en cada universidad determinan el sesgo del plan de cada una.

## ¿Qué lenguaje de programación?

Otro de los problemas habituales a la hora de implementar un plan de estudios de informática es elegir qué lenguaje o lenguajes de programación emplear. Parece razonable que esta decisión se tome fundamentalmente por razones pedagógicas, sin embargo otros muchos factores afectan y han afectado esta decisión.

Durante cierto tiempo muchas universidades emplearon Pascal como lenguaje para enseñar programación. En otras asignaturas, típicamente las relacionadas con los sistemas operativos, se empleaba C por ser el más usado en la implementación de sistemas operativos, los profesores de asignaturas de control o tiempo real preferían Ada, etc. Estas aproximaciones eran criticadas desde la industria por ser lejanas a la realidad del mercado. Es muy importante que aprendan COBOL clamaban hace 15 años...

A mi entender, la *industria* parece no darse cuenta de que van por detrás de la universidad y no por delante en estas cuestiones (en contra de lo que creen). Me explico, hace algunos años para mucha gente de la industria en la universidad se explicaban cosas *poco útiles* cuando no enseñaba COBOL. Curiosamente, como los alumnos en las universidades aprendieron otras cosas, en particular C, cuando llegaron a la industria empezaron a usar lo que mejor conocían. Así, hicieron uso de C, por ejemplo, incluso para hacer programas de gestión empresarial, para lo que no está especialmente dotado.

Hoy la situación sigue siendo similar. Ahora desde la industria se reclama, por ejemplo, que se use Java como lenguaje de programación. Mejor dicho, se pide que se les enseñe Java y sus implementaciones de todas las tecnologías básicas y casi nada más. Sé que para muchos la idea de un lenguaje interpretado con fuertes conexiones con la red puede parecer algo novedoso, pero la verdad es que había muchas cosas similares y mejores antes, simplemente Java tuvo la suerte de nacer entre los brazos de una multinacional.

## Nuevo factor: la industria

Además se añade un nuevo factor, la industria ha crecido mucho y ahora hay jugadores muy poderosos: Microsoft, SUN, etc. que entienden perfectamente que lo que aprendan los futuros ingenieros será un factor de mucho peso en lo que se utilizará en las empresas en los años venideros. De ahí que muchas empresas hagan muchos esfuerzos para colocar sus productos en las universidades.

Microsoft parte de su conocida ventaja en sistemas operativos de los equipos de sobremesa, que son los que fundamentalmente equipan las aulas de prácticas por razones fundamentalmente de precio en las universidades. Hasta hace muy poco era complicado encontrar aulas de ordenadores compatibles con sistemas operativos que no fuesen de Microsoft. La resistencia ha sido muy fuerte a la aparición de aulas con GNU/Linux u otros sistemas operativos libres. Las multinacionales informáticas han creado licencias *campus* para tratar de convencer a las universidades de su uso. Han realizado *donaciones* de software con la misma idea.

Mi pregunta fundamental es: ¿Debería obligar la administración a utilizar tecnologías neutrales, esto es libres, en la docencia en tecnologías de la información? A mi entender sí, por dos razones fundamentales. La primera menos importante: por precio. Las licencias de campus son muy baratas comparadas con los precios individuales, pero aún así hay que pagarlas y las donaciones sólo se producen al subconjunto limitado de universidades de mucho prestigio con la idea de que arrastren a las demás.

La segunda es la más importante desde mi punto de vista: las administraciones públicas deberían forzar el uso de tecnologías no propietarias para no colaborar en el mantenimiento de monopolios en ningún campo de estas tecnologías. Aceptar que se emplee como lenguaje Java, o que los únicos sistemas operativos que vean los alumnos sean los de la familia Microsoft Windows, me parece peligroso desde el punto de vista social.

Curiosamente, para alguien que ha defendido la vertiente más ingenieril de la informática, son los *sectores* más partidarios de la fuerte formación matemática los que más comprenden la necesidad de usar tecnologías no propietarias. Quizás la no patentabilidad de las matemáticas ha llevado a esa comunidad a una evolución más abierta que a la de las supuestamente más dinámicas ingenierías.

## Deontología profesional

Otro punto de interés en la formación de los informáticos, y que se deja muchas veces de lado, es la formación sobre la informática como rama del saber y sobre sus componentes sociales. La impartición de este tipo de conocimientos suele hacerse alrededor de la asignatura de deontología profesional. Esto no es exclusivo de la informática, casi todas las titulaciones universitarias imponen ciertos créditos obligatorios de deontología profesional.

Las asignaturas sobre deontología en informática tienen en mi opinión un perfil muy limitado, que reconozco sesgada por la implementación de las asignaturas que conozco. En general se centran en una única parte de los problemas a los que se enfrenta un informático en su vida profesional: el uso de información confidencial, bien de los usuarios de los servicios que desarrollan, de sus compañeros o de las empresas para las que trabajan. También abordan, como no podría ser menos, las prácticas éticas, por ejemplo no realizar programas incorrectos intencionadamente, se introducen discusiones sobre la necesidad o no de colegios profesionales (esto daría para otro artículo), se habla de las prácticas de contratación y relación con los compañeros y competidores, y en general se establecen los derechos y deberes legales de la profesión informática.

Sin embargo, en mi opinión es deseable introducir a los alumnos en otros asuntos relacionados con la informática, pero que son de caracter más general. Creo que es importante que se enfrenten a los problemas sociales de la informática. En particular me parece imprescindible que conozcan la filosofía subyacente en el modelo de desarrollo del software libre. No es aceptable que la mayoría de

los alumnos de informática salgan de las aulas pensando que la única forma de desarrollar software, o de ganar dinero desarrollándolo es mediante la venta de licencias propietarias. Es necesario que se aborde el análisis de las licencias de software, que se estudien sus implicaciones.

De igual forma es necesario que en las aulas se aborden las cuestiones relacionadas con la propiedad intelectual, de las que yo creo que el software libre es un subconjunto. La informática está estrechamente relacionada hoy en día con la producción y distribución de contenidos multimedia. Es necesario que los informáticos discutan y razonen sobre las formas en que se van a realizar esas funciones y sus responsabilidades sociales.

Las prácticas profesionales de un informático no deben limitarse a su relación con la información confidencial, la privacidad de los datos, etc. Es importante que sea consciente de su papel en la cadena de producción intelectual y que decida con conocimiento qué tipo de herramientas desarrolla o utiliza en su vida profesional.

## Los otros informáticos

Hasta aquí he usado la formación de los informáticos como hilo conductor del artículo. Soy consciente de que la profesión informática a día de hoy está llena de gentes de procedencias muy diversas. Así, además de ingenieros informáticos, hay ingenieros de otras ramas, licenciados, etc. Eso sí, lo dicho sobre el uso de herramientas informática libres se aplica igualmente a su formación.

Asignaturas relacionadas con la gestión de la propiedad intelectual en general y con el modelo de desarrollo del software libre en particular creo que deberían ser obligatorias en los estudios relacionados con las tecnologías de la comunicación y muy recomendables (quizás en formato de asignaturas de libre elección/configuración) en otras titulaciones.

Otra parte de los profesionales de la informática son los autodidactas. A ellos les recomendaría que, siguiendo sus instintos, trataran de ver más allá de las herramientas comerciales de moda y que apostasen por el software libre como camino de especialización.

Por último, como profesor no puedo dejar al resto de los usuario sin deberes: no vale con ser simples usuarios de GNU/Linux, hay que entender, por ejemplo, qué papel juega la licencia GPL y el modelo de desarrollo de software libre en general.

Como siempre BarraPunto sigue siendo un buen sitio para discutir estos asuntos. Así, si quieres leer más sobre estos temas puedes intervenir en muchas de las discusiones que se celebran al respecto.

# La Administración Pública

Una de las líneas argumentales más sólidas en el pensamiento relacionado con el software libre es el papel de las administraciones públicas. A diferencia de otros campos, cuando algunas administraciones financian el desarrollo de software, la titularidad sobre los derechos de ese software quedan en manos de los contratistas que lo desarrollan. Muchos entonces clamamos al cielo y nos preguntamos, tal y como se titula el primer ensayo de este capítulo, *¿Qué se hace con mi dinero?*. A fin de cuentas, las administraciones públicas tienen una serie de obligaciones contraídas con sus ciudadanos y, como se defenderá en este artículo, van más allá de la compra de software y servicios relacionados.

*PADREs y otros parientes oficiales* presenta uno de los grandes problemas con el que nos encontramos los usuarios de sistemas operativos libres: pagar nuestros impuestos. Y ya no es que sea más o menos doloroso tener que hacerlo. El problema se agrava, ya que el famoso programa PADRE sólo está disponible para sistemas Microsoft Windows, lo que hace que, sin lugar a dudas, esto de cumplir con el fisco sobrepase el umbral del dolor. Aprovechando la ocasión para exigir un PADRE para todos, también se ofrecen argumentos de por qué hacerlo con software libre tiene mucho sentido.

*CEE: Ciudadanía Electrónica Europea* es un ensayo de más altos vuelos en el que se presentan varias propuestas para mejorar la democracia en la que vivimos. Al fin y al cabo, seguimos teniendo una democracia que se acuñó a principios del siglo pasado y que no ha evolucionado mucho en sus formas y métodos... ¡y eso que nadie discutirá lo mucho que el mundo ha cambiado! Entre esos cambios, cómo no, están los del campo de las tecnologías de la información. En este artículo se pueden encontrar unas cuantas ideas al respecto.

Hace pocos siglos nadie hubiera apostado por tener a nuestra disposición trabajos tan complejos y elaborados como una enciclopedia o un diccionario. Desde hace relativamente poco en comparación con la historia de la humanidad, los hemos tenido a nuestra disposición gracias a la organización de generalmente unos pocos que han dedicado mucho tiempo y esfuerzo a que esto sea posible. Pero lo que hace tiempo supuso una gran revolución, hoy se ha quedado pequeño. Un ejemplo de lo que se comenta lo podemos encontrar en *El Diccionario de la Real Academia de la Lengua*. Y es que la Real Academia de la Lengua podría aprender muchas cosas del sistema en que se genera y difunde el software libre. La idea es muy simple: utilicemos las herramientas y métodos de trabajo colaborativo que han hecho de muchos proyectos de software libre un éxito y apliquémoslas

en la (re)generación del Diccionario. Es probable que de esta forma los objetivos originarios de la Real Academia de la Lengua se vean potenciados de manera indiscutible.

# ¿Qué se hace con mi dinero?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 17, pág. 12-13, Marzo de 2002

En general, me preocupa cómo se gasta el dinero de mis (y tus) impuestos. Sí, soy un bicho raro de esos que paga su declaración de la renta más a gusto si cree que el dinero se emplea bien. O al menos, si cree que no se emplea mal...

Pero claro, de grandes partes del gasto público no entiendo nada. Así que en esos casos me es difícil decidir si las cosas se hacen como a mi me parece bien, porque no sé cómo me parece bien. Pero de algunas partidas sí entiendo. O al menos, creo que entiendo, y eso me basta para atreverme a escribir sobre ellas. Y en particular, lo has adivinado, me preocupa cómo se gastan las administraciones públicas mi dinero cuando hablamos de informática, y especialmente de software. Y cuando miro un poco, no me gusta lo que veo. Los responsables de todas las administraciones a las que pago (salvo contadas excepciones) parecen no haberse enterado de que hay más de una forma de hacer las cosas. Parecen ignorar que se puede gastar el dinero público que se dedica al software de una forma mucho más adecuada a las necesidades, y proporcionando un beneficio social mucho mayor. Y me preocupa sobre todo que, debido a esta ignorancia, todas las administraciones, desde mi ayuntamiento hasta la Comisión Europea, estén dejando pasar una oportunidad única. En lugar de promover un cambio que nos conviene a todos, son un factor de inercia. Naturalmente, me estoy refiriendo a la aparente ignorancia de casi todos los estamentos públicos sobre todo lo que tiene que ver con el software libre.

## La informática y la administración

Las administraciones públicas actúan en el mercado del software al menos de tres formas:

- Comprando programas y servicios relacionados con ellos. Las administraciones, como grandes usuarios de informática, son un actor fundamental en el mercado del software.
- Promoviendo de diversas formas el uso (y la adquisición) de ciertos programas en la sociedad. Esta promoción se hace a veces ofreciendo incentivos económicos (desgravaciones fiscales, incentivos directos, etc.), a veces con información y recomendaciones, a veces por *efecto ejemplo*.
- Financiando (directa o indirectamente) proyectos de investigación y desarrollo que están diseñando el futuro de la informática.

En cada uno de estos ámbitos el software libre puede presentar ventajas interesantes tanto para la administración como para la sociedad en general. Sin

embargo, raramente se considera apoyar, fomentar o siquiera tener en cuenta soluciones libres en alguna iniciativa concreta. Y esto nos está perjudicando a todos.

## ¿Cómo se satisfacen mejor las necesidades de la administración?

Las administraciones públicas son grandes consumidores de informática. En lo que al software se refiere, compran habitualmente tanto productos de consumo masivo (*off-the-self*[1]) como sistemas a medida.

En el primer caso, lo habitual es considerar sólo software propietario. La cantidad de recursos públicos que se gastan los ayuntamientos, las comunidades autónomas, la administración central y las administraciones europeas en comprar licencias de Microsoft Windows, Microsoft Office u otros productos similares es ciertamente considerable. ¿Por qué no se consideran, siquiera parcialmente, soluciones libres? ¿Realmente en ningún caso se puede usar GNU/Linux con GNOME o KDE y OpenOffice.org, por ejemplo? Y aunque técnicamente fuera imposible hacerlo ahora, ¿no deberían estar tomándose medidas desde hace tiempo para preparar el futuro?

Por ejemplo, con una fracción de lo gastado en dos o tres productos propietarios *estrella* por todas las administraciones europeas (o incluso las de casi cualquier estado), se podría promover un concurso público para que una empresa (o dos, o tres, o cuatro) mejorasen y adaptasen los programas libres ahora disponibles para que en el plazo de uno o dos años estuvieran listos para su uso masivo al menos para ciertas tareas típicas. Imaginad por ejemplo un esfuerzo coordinado, a nivel nacional, o a nivel europeo, para que todas las administraciones participasen en un consorcio que se encargase de la gestión de estos concursos. En poco tiempo habría una industria europea especializada en realizar estas mejoras y estas adaptaciones. Y las administraciones podrían elegir entre las tres o cuatro distribuciones libres producidas por esta industria. Para fomentar la competencia se podría recompensar económicamente a cada empresa según la cantidad de administraciones que eligiesen usar su distribución. Y todo el resultado de esta operación, al ser software libre, estaría también a disposición de empresas y usuarios individuales, que en muchos casos tendrían necesidades similares a las de las administraciones.

En el caso del software hecho a medida, el proceso normalmente pasa por contratar con una empresa los programas necesarios bajo un modelo propietario. Todo el desarrollo realizado a petición de la administración es propiedad de la empresa que lo desarrolla. Y normalmente la administración contratante queda atada a su proveedor para todo lo que tenga que ver con mejoras, actualizaciones

[1] Nota del editor: literalmente *de la estantería*, como si el software se pudiera adquirir en un supermercado al igual que cualquier otro producto de consumo masivo.

y soporte, en un círculo vicioso que dificulta mucho la competencia y ralentiza el proceso de modernización de las administraciones públicas. Lo que es peor, en muchos casos el mismo programa es vendido una y otra vez a administraciones similares, aplicando en cada nuevo caso los costes que habría supuesto hacer el desarrollo desde cero. Por ejemplo, ¿cuántos ayuntamientos con necesidades informáticas similares pagan por un software que ya ha sido pagado por otro? ¿Por qué no se consideran soluciones libres al menos en parte de estos ámbitos?

Un consorcio de administraciones públicas con necesidades de un cierto software a medida podría exigir que el resultado obtenido fuera software libre. Esto permitiría que otras administraciones se beneficiasen también del trabajo, y a medio plazo estuvieran interesadas en colaborar en el consorcio para que se tuvieran en cuenta sus necesidades peculiares. Al ser el software resultante libre, no habría obligación de contratar las mejoras y adaptaciones al mismo proveedor, introduciendo de esta forma competencia en ese mercado (que hoy por hoy es casi cautivo). Y en cualquier caso, el coste final para cualquiera de las administraciones implicadas no sería nunca mayor que si se hubiera utilizado un modelo propietario. ¿Qué se puede perder?

## Cuidado con lo que se promueve

Cuando una entidad pública promueve cualquier tipo de plan que incentiva el gasto en informática es conveniente estudiar con mucho cuidado sus implicaciones. Por ejemplo, no es muy buena idea promover el uso de Internet en la sociedad de una forma que fomente hasta la exageración un determinado monopolio, que a la larga será perjudicial para todos (salvo para la empresa monopolística, claro).

Éste es desgraciadamente un caso muy habitual. Con buenas intenciones, la clase política decide invertir recursos en ayudar a un determinado sector a que se modernice. Se crean programas de exención de impuestos por determinadas compras, se realizan adquisiciones masivas de hardware y software para escuelas, etc. Pero raramente se diseña con cuidado la iniciativa, teniendo en cuenta todas sus implicaciones. Por ejemplo, no se suele considerar que la misma cantidad de dinero se puede usar en comprar una cierta cantidad de licencias de un programa propietario, o en adquirir una copia de uno libre, y contratar soporte o adaptaciones para él. Así, en lugar de modernizar la sociedad, se fomenta la compra de ciertos productos, desincentivando la de otros que habrían producido más beneficios en la sociedad.

¿No es razonable estudiar el gran potencial que tiene el software libre en este tipo de iniciativas? Por ejemplo, imaginemos por un momento que parte de la cantidad destinada a un programa de informatización de escuelas se dedica a crear una distribución de GNU/Linux adaptada a las necesidades de la docencia en enseñanza primaria. Y con el resto de los recursos, se contrata soporte para que el software sea mantenido en esas escuelas, de forma que no sea un simple *software florero*, sino que realmente haya gente encargada de su correcto funcionamiento.

Así se cubren no sólo las necesidades del sistema educativo, también se genera un mercado para empresas, habitualmente de ámbito local, capaces de ofrecer servicios de mantenimiento. Y por supuesto, se deja completamente abierto el camino al futuro: el software no quedará obsoleto en pocos años, obligando a comenzar desde cero, sino que se podrá ir actualizando incrementalmente, año a año, manteniendo los beneficios del programa con una inversión similar.

## Si pagamos, los resultados han de estar a nuestra disposición

El caso del dinero que las entidades públicas dedican a fomentar la investigación es especialmente sangrante. Con nuestros impuestos se está construyendo gran cantidad de software del que no nos vamos a beneficiar ni siquiera indirectamente. Habitualmente, los programas públicos de fomento de la investigación y desarrollo financian total o parcialmente proyectos que crean programas sin atender a qué derechos va a tener el público sobre ellos. En muchos casos los resultados, sin un plan adecuado de comercialización, simplemente quedan en algún cajón, cubriéndose de polvo. En otros, las mismas personas que financiaron un programa vía impuestos acaban pagándolo de nuevo si lo quieren usar (ya que tienen que adquirir licencias de uso).

¿No sería más razonable que los programas resultado de la investigación financiada con fondos públicos estuvieran disponibles para el público? Especialmente en los casos donde la investigación es precompetitiva (lo más habitual en los casos de financiación pública) esto permitiría que la industria europea del software se beneficiase enormemente. Donde una empresa sólo ve un resultado de imposible comercialización, otra puede ver una oportunidad de negocio. Así, por un lado, se maximizan los resultados de los programas de investigación. Y por otro, se favorece la competencia entre las empresas que quieran utilizar los resultados de un proyecto, ya que todas ellas competirán a partir de los mismos programas resultado del proyecto.

Este modelo no es nuevo. En gran medida, es el que ha permitido el desarrollo de Internet. Si las administraciones públicas exigen que los resultados de la investigación realizada con sus fondos sean distribuidos en forma de programas libres, podemos ver aparecer casos como el de Internet en cada esquina. O bien los resultados de esas investigaciones son malos o inútiles (y en ese caso debería replantearse la forma de selección de los proyectos), o bien la dinamización que supondría dejarlos listos para que cualquier empresa pueda convertirlos en producto permitiría desarrollos sencillamente impredecibles.

## Aún no se puede concluir...

Normalmente al final de un artículo se escriben unas cuantas conclusiones. Pero en este caso no lo voy a hacer. Porque sólo el tiempo permitirá escribirlas.

Hoy por hoy, lo único que podemos hacer es exigir, como contribuyentes que somos, que se de una oportunidad al software libre en la administración pública. Tenemos mucho que ganar, y es difícil perder algo. En el mejor de los casos, dentro de unos años se percibirá una revitalización del sector informático, con nuevas empresas por toda la geografía, con mejores servicios para los ciudadanos, y con un gran efecto multiplicador. En el peor caso, probablemente las cosas estarán mejor que ahora...

Pero para conseguir estos beneficios harán falta gestores públicos que se animen a hacer las cosas de una forma nueva. Políticos y funcionarios que se atrevan a plantearse cuál es la mejor manera de hacer las cosas. Y empresas y empresarios que estén a la altura del reto.

Así pues, la cuestión es: ¿tenemos de esto?

# PADREs y otros parientes oficiales

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

Ha llegado mayo, la primavera, la alergia y la hora de que los españoles ejerzamos el derecho y la obligación de pagar los impuestos directos que nos corresponden. Hasta aquí todo bien (o mal si sufres de alergia al polen). Los problemas con la declaración de impuestos comienzan cuando uno intenta hacerlo por métodos informáticos. Bueno, realmente tienes problemas si eres usuario GNU/Linux, si eres usuario del reconocido monopolio americano tienes muchos menos problemas (únicamente los propios de usar un sistema operativo que no es libre).

Para los usuarios de Microsoft Windows, la Agencia Tributaria (AEAT) ha desarrollado un programa, probablemente el software español más extendido, que se llama PADRE y que sirve para completar la declaración del impuesto de la Renta de las Personas Físicas (o IRPF). Pero ¿qué ocurre si no eres usuario de dicho sistema operativo? Pues resumidamente: nada. No dispones de ningún programa, y ni siquiera hay visos de que vayas a poder tenerlo. ¿Cómo es posible que exista un monopolio de este tipo y que nadie proteste?

## Sobre monopolios informáticos

¿Qué opinarían los fabricantes de bolígrafos si sólo se pudieran rellenar los impresos de hacienda con una marca determinada de bolígrafo? (especialmente los de las marcas no elegidas). ¿Qué dirían los nacionalistas, de cualquier nación, si encima el bolígrafo fuese americano...? Y qué no dirían los banqueros si ese bolígrafo además no pudiera abrirse y no fuese la primera vez que un bolígrafo hace cosas raras ¡manejando sus cuentas!

Sin embargo, como se trata de una herramienta informática pues todo el mundo se sorprende cuando les explicas que en tu ordenador no lo puedes usar: "ya está el raro ese del *Linux*". Pero es todavía peor cuando explicas que lo que te gustaría es que te den el código fuente para intentar portarlo, o que alguien más habilidoso lo haga, o para hacerlo entre todos... Entonces es cuando te miran con cara de "ya está el listo que no quiere pagar impuestos", "éste fijo que es un cracker que se va a librar y además va a cotillear mi declaración".

Simplemente no se entiende, ni el problema: que la administración use software propietario; ni la propuesta de solución: que use software libre.

El año pasado desde BarraPunto, y desde otras organizaciones alrededor del software libre, se animó a los usuarios a enviar un mensaje de correo electrónico a la AEAT pidiendo una versión para GNU/Linux de dicho programa. La respuesta estándar se recibía al cabo de un par de días, indicando que se había

decidido realizar versiones del programa únicamente para el sistema operativo más usado. Lo que me resulta curioso, porque hace algunos años tenían al menos dos versiones, una para PC compatibles y otra para los Apple. ¿Será que Apple ha perdido fuerza o que Microsoft la ha ganado? ¿Será que detrás de GNU/Linux no hay ninguna empresa con fuerza suficiente para reclamar el mismo trato que Apple en su día? ¿O será que los usuarios no hacemos la suficiente fuerza?

En la campaña proponíamos también responder a ese mensaje con la petición de la publicación del código, que no debería ser problema, ya que especifican que es software de *libre disposición*. Sin embargo, a ese segundo mensaje se obtenía el silencio como única respuesta. ¿Cuál puede ser el problema de la administración para liberar ese programa?

El mismo tipo de respuesta, básicamente que de otra cosa que no fuese Microsoft Windows no sabían nada, se obtenían para las consultas sobre cómo presentar la declaración por vía telemática. Es más, el año pasado una amable telefonista estaba empeñada en que me instalara el Microsoft Explorer en mi máquina en vez del Netscape "que a otra gente también le ha dado muchos problemas con los certificados". La buena mujer no lo podía entender: "No puede ser que no se lo pueda instalar si tiene tanto espacio en el disco duro como dice". Obviamente ni se le pasaba por la cabeza que alguien pudiera tener otra cosa que Microsoft Windows, y aunque se lo explicases tampoco lo entendía. La administración pública te recomienda un producto comercial y además te da soporte sólo para ése en concreto.

Este año la iniciativa en favor de la liberación del PADRE ha partido desde HispaLinux, esperemos que siendo cada vez más los usuarios de GNU/Linux y de software libre en general consigamos algún día la liberación de nuestro PADRE.

Por cierto, en 1996 el programa necesitaba un mínimo de 640Kb de memoria y 2MB de disco (todavía conservo el disquete). La versión de este año necesita un mínimo de 32MB, aunque recomienda 64MB, de memoria y 12MB de espacio en el disco duro, y eso que se supone que la declaración se ha simplificado bastante, o al menos eso se afirma desde la propia AEAT. ¿Será por eso que no nos quieren enseñar el código fuente? Igual tenemos algún simulador de vuelo en nuestro programa... No es broma, no sería el primer programa comercial que incluye software no deseado en la distribución, lo que nos lleva a la siguiente pregunta.

## ¿Qué hay en el ejecutable del PADRE?

No es sólo un problema de preferencia de sistema operativo, ni siquiera de moralidad sobre el destino de los dineros públicos. Se puede observar el problema desde otros ángulos. Por ejemplo, el derecho a la intimidad. ¿Qué hay realmente en ese programa en el que introducimos datos tan sensibles como nuestros ingresos y nuestros gastos?

Algunas sociedades, por ejemplo las anglosajonas, están muy sensibilizadas con los problemas de la intromisión del estado en sus vidas privadas en general,

por la información que el gobierno tiene de los ciudadanos, etc. Eso hace, por ejemplo, que sean muy reacias a tener instrumentos como un documento nacional de identidad.

En España tenemos una preocupación mucho menor por estos problemas. Pero deberíamos empezar a planteárnoslo: descargamos un binario de la AEAT (a través del servicio de una multinacional americana que se llama Akamai), rellenamos nuestros datos, desde el número de hijos al nivel de minusvalía pasando por todos nuestros datos económicos. Nos volvemos a conectar a la AEAT y enviamos esos datos. La AEAT nos asegura que esos datos van cifrados y que sólo se envían a ellos. ¿Alguien puede asegurar que ese programa sólo envía la declaración? ¿qué no envía las simulaciones que has hecho para ver cómo conviene más hacerlo?. Pero no hace falta pensar únicamente qué hará con los datos introducidos en el programa. ¿Alguien puede asegurar, por ejemplo, que no envía una descripción de los programas instalados en el ordenador a algún sitio? ¿O que no lo hará dentro de unos meses?

Por supuesto, no estoy insinuando que el programa PADRE lo haga, al contrario, creo que no lo hace. Sin embargo no puedo estar realmente seguro de lo que hace porque no puedo mirar el código fuente, ni puedo pedirle a un amigo con más experiencia en estas cosas que lo haga. ¿Por qué no disponemos del código fuente de ese programa? La normativa legal con la que está confeccionado es pública (la legislación tributaria), los cálculos que tiene que realizar son bastante sencillos (sumas y porcentajes básicamente). No parece un secreto que ponga en peligro la seguridad nacional.

## El problema se extiende

Por desgracia, no es el único ámbito de la administración pública donde está generalizado el uso de los productos y formatos propietarios. Por ejemplo, muchas universidades lo imponen como estándar, y exigen a sus trabajadores que toda la burocracia interna, desde los programas de asignaturas a las notas, se realicen en formatos propietarios.

Usando el mismo ejemplo anterior ¿Se atreverán las universidades a exigir a los alumnos que realicen los exámenes con una determinada marca de bolígrafo? ¿O a que entreguen la documentación de su matrícula en un determinado formato electrónico propietario? En el campo de la enseñanza han existido grandes polémicas con casos mucho más pequeños. Recuerdo las campañas para que los colegios privados no pudiesen vender los libros que ellos mismos recomendaban. ¿Cómo se puede aceptar que en las universidades se fuerce a utilizar un determinado software?

Otro caso, el recién creado Ministerio de Ciencia y Tecnología, en un alarde de modernidad ha creado su propio “PADRE” para formalizar la petición de ayudas a la investigación. Así, en la última convocatoria de proyectos PROFIT es obligatorio entregar la documentación en un programa para Microsoft Windows.

No es un formulario, ni un formato como ocurría hasta ahora; es un ejecutable para una plataforma determinada, por supuesto no libre[1].

Desde luego no hay justificación técnica para hacer esto. Tampoco creo que exista *maldad* detrás de estas decisiones. De nuevo no existe conciencia del problema. ¿No sería mucho más fácil un simple formulario HTML para la mayor parte de las interacciones con la administración?

¿No deberían los organismos públicos fomentar el uso de software libre? Es más, ¿no deberían estar ellos mismos obligados a que todo el software desarrollado con dinero público sea público? ¿Cómo puede ser que un ministerio pague a una empresa por un desarrollo y sea la empresa la que se quede con el código? O peor todavía, ¿cómo puede el estado tener el código y no compartirlo con los ciudadanos que lo han pagado?

El gobierno americano tiene, al menos en algunas áreas, esa norma que, por otra parte, genera empresas y tecnología. Un ejemplo bien conocido es la empresa ACT que comercializa el compilador de Ada (GNAT) más usado, y que nació como resultado de un contrato con las Fuerzas Aéreas estadounidenses para desarrollar ese compilador. El software, al haber sido pagado con dinero público tenía que ser libre, sin embargo ACT se creó para explotar ese producto libre. Así, las empresas, entre ellas la española CASA (Construcciones Aeronáuticas S.A.), o la misma Agencia Espacial Europea (ESA), pagan por el soporte, por la resolución de problemas, o por mejoras *a medida*.

Tengo entendido que esta política de que el software desarrollado con dinero público debe ser de dominio público o libre, no es exclusiva de la aviación militar americana. Al parecer es bastante general en la administración americana. En Europa, sin embargo, es al contrario. Las empresas que participan en proyectos con financiación europea (los ESPRIT e IST) se reservan no sólo los derechos de explotación del software desarrollado, sino el código mismo. Es decir, con dinero público estamos financiando desarrollos privados. ¿Cómo es esto posible? Fundamentalmente porque dejamos que suceda.

## ¿Qué se puede hacer?

Pues básicamente lo mismo de siempre, hacer oír nuestra voz. En este caso enviando mensajes educados solicitando versiones libres del programa PADRE a la Agencia tributaria: `mailto:padres2@aeat.net` y protestando cada vez que en una administración pública nos obliguen a entregar documentación en un formato propietario. Y sobre todo explicar el problema, explicar que las cosas no tienen necesariamente que ser así, que no son así en casi ningún otro entorno y que además no es bueno que lo sean.

---

[1] Se puede encontrar más información al respecto en: `http://www.mcyt.es/profit/cuestionario.htm`

# CEE: Ciudadanía Electrónica Europea

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

Hace mucho, mucho tiempo, en un país muy, muy lejano, los ciudadanos de una de sus ciudades discutían cuál era el mejor sistema para su gobierno. Algunos decían que era deseable que gobernasen los mejores, el problema era dilucidar quiénes eran: si los más listos, los más ricos, los más fuertes... Otros opinaban que la estabilidad de un sistema hereditario era muy deseable: pero quiénes empezaban esas dinastías y qué poderes asumía cada una, era objeto de controversia... Otros opinaban que era necesario contar con los dioses y sus representantes en la tierra: el debate esta vez era que había muchos dioses y muchos más sacerdotes... La discusión fue muy larga, sufrieron numerosas guerras y probaron sistemas diversos. Al final decidieron que lo mejor era que los ciudadanos decidiesen individualmente bajo el principio de *un ciudadano un voto*, y que todos acatasen la decisión mayoritaria. A ese modelo se le suele conocer como *democracia*.

Hoy los ciudadanos de la Unión Europea estamos pensando en darnos una constitución que reconozca ese mismo principio como base de nuestro gobierno y de paso generar una ciudadanía europea. Me temo que, desgraciadamente, en su implementación no se van a tener en cuenta las cuestiones tecnológicas que harían la implementación de este principio más cercana a la intención inicial.

## La democracia representativa

La idea de los ciudadanos de aquel país se exportó a muchos otros países, sin embargo, como estos otros países eran grandes y las comunicaciones en aquellos tiempos no eran muy buenas (caminos y carros de bueyes fundamentalmente) decidieron hacer algunos *ajustes* a la idea básica. El más extendido fue que en vez de que todos los ciudadanos tuviesen que viajar para reunirse en una asamblea multitudinaria eligieran periódicamente unos representantes en quienes delegarían su voto. A este sistema se suele denominar *democracia representativa* y es el más extendido entre las, a mi juicio, llamadas democracias modernas o parlamentarias.

Este ajuste tan pequeño, pasar de la democracia directa a la democracia representativa, ha significado un enorme cúmulo de problemas: cómo se elegían los representantes (por regiones, castas, sexos, ...), cuánto duraba su mandato, cómo se les revocaba si las decisiones que tomaban estos representantes dejaban de gustar a sus electores, etc. Con todos estos problemas, durante los últimos dos siglos este sistema parece ser el que ha producido mejores resultados.

Sin embargo, aún reconociendo que puede ser *el menos malo de los sistemas* creo que su implementación actual se ajusta poco a la idea original. Son

varias las causas de la perversión del sistema, en primer lugar el poder que en general detentan estos representantes ha hecho muy deseable su control desde los diferentes sectores de la sociedad. De esta forma, han aparecido los *lobbys* (grupos de interés), reconocidos legalmente o no, que se encargan de defender intereses corporativos, de casta, etc. ante los representantes. Como consecuencia, los ciudadanos *de a pie* pierden interés por el funcionamiento del sistema que se considera viciado.

Otro problema de la democracia representativa es la aparición de una nueva casta, la de esos representantes. Es curioso observar, por ejemplo, cómo existen familias en España en las que hay varios diputados o senadores en diferentes generaciones o ¡incluso en la misma generación! Un estudio estadístico diría que ese hecho debería ser prácticamente imposible teniendo en cuenta el número de familias españolas y el número de escaños. Esto indica que realmente la democracia representativa genera una nueva casta, la de los políticos. Como luego argumentaré, es un impedimento más a la reforma del sistema, además de ser otra causa del distanciamiento de los ciudadanos de la política, a la que acusan de ser únicamente la plasmación de los intereses de los *lobbys*, conseguida a través de la casta de los políticos.

## Otra implementación es posible

Yo creo que la tecnología nos permitiría hoy volver a un sistema político más parecido a su concepción ideal: gobierno directo de los ciudadanos. ¿Cómo se podría implementar un sistema de democracia directa en una sociedad moderna como la nuestra? Por supuesto, mi propuesta se basa en el uso de la tecnología, por ejemplo usando dos de los sistemas de comunicación más extendidos hoy en día: la televisión e Internet. ¿Qué impide que tengamos todas las semanas (o todos los días) un programa en televisión para explicarnos las diferentes propuestas de ley? ¿Qué impide disponer de un sistema de voto electrónico para aquellos ciudadanos interesados en participar en la vida política de su comunidad? De hecho, con Internet creo que nos basta, pero quizá todavía no esté lo suficientemente extendida.

Seamos por otra parte pragmáticos, los grupos de interés no van a desaparecer. Aparecerán también en el nuevo sistema, luego es necesario articular contrapesos, por ejemplo igualando los tiempos de explicación a favor y en contra de las propuestas en la televisión, fomentando la aparición de grupos de interés como las diferentes organizaciones no gubernamentales (ecologistas, asistenciales, etc.).

Siguiendo con la línea pragmática, es también probable que una gran parte de la población no esté interesada en todos los temas que se abordan hoy en día en un parlamento, o no querrán seguir los programas de televisión en los que se exponen los distintos puntos de vistas, o considerarán que su opinión no está lo suficientemente bien formada sobre un determinado tema (por ser demasiado técnico, por ejemplo), etc. Para esas cuestiones nada impide que se siga usando

el sistema de representación, o una versión evolucionada que denominaremos *delegación de voto*.

Gracias al soporte que nos proporciona la extensión de Internet, es relativamente sencillo implantar un sistema flexible de delegación de voto para los ciudadanos. Con este sistema un ciudadano podría delegar su voto para las votaciones relacionadas con un determinado tema, por ejemplo con las inversiones en el Plan Hidrológico en Greenpeace, pero podría retener su voto para opinar sobre aspectos concretos, por ejemplo para votar sobre el regadío que afecta directamente a su pueblo. Es decir, la delegación de voto en un sistema de voto electrónico no tiene por qué tener un periodo fijo como ocurre en nuestros sistemas parlamentarios. La delegación no tiene tampoco por qué ser unidimensional, se podría por ejemplo delegar voto por temas, no tiene tampoco por qué ser una delegación unipersonal, se podría delegar el voto en organizaciones reconocidas, como por otra parte se hace hoy con los partidos en España, donde tenemos listas cerradas de candidatos. Así, por ejemplo un ciudadano podría delegar su voto en Greenpeace para los asuntos relacionados con el Plan Hidrológico y en los partidos políticos tradicionales para otros temas.

Esta idea de la democracia directa con sustrato tecnológico no es nueva. Ross Perot, aspirante independiente (en el sentido de no ser apoyado ni por el partido Demócrata ni por el Republicano) a la presidencia de los EEUU en los años noventa proponía ya hace tiempo que el modelo de democracia representativa estaba pervertido. Concretamente una de las obsesiones de Ross Perot era precisamente el enorme poder que tienen los *lobbys* de Washington, muy probablemente porque sabía el dinero que como dueño de EDS tuvo que pagarles para defender sus intereses ante el gobierno federal de los EEUU.

## Las aportaciones del software libre

Ésta es una revista que teóricamente habla de GNU/Linux, ya es bastante raro que el tipo éste de las barbas[1] que firma el artículo escriba siempre sobre cosas raras como los derechos de autor, las patentes software, etc. pero eso por lo menos tiene algo que ver con el software libre que es el fundamento de GNU/Linux, pero ¿qué tiene que ver este ensayo barato sobre la democracia directa? Seguro que como lector te preguntas esto, pues bien, desde luego ese sistema electrónico que soporte un sistema como el propuesto, para mí tiene que reunir algunas características. Una de las más importantes, desde mi punto de vista, es que debe estar a disposición del público para que se garantice su fiabilidad, para lo cual el mejor mecanismo es que sea un sistema bajo licencia de software libre.

[1] Nota del editor: la revista que originalmente publicó este artículo tiene la costumbre de poner una fotografía tamaño carné de los autores. Y Vicente Matellán, como muchos de los personajes ilustres en el mundo del software libre, tiene barba.

En la actualidad algunas empresas, por ejemplo la española Indra[2], venden sistemas de voto electrónico que se han usado en elecciones políticas tan grandes como las últimas presidenciales de Brasil, pero estos sistemas únicamente sustituyen a la papeleta tradicional para hacer más fácil el recuento. Otras empresas venden sistemas de voto con terminales individuales que se pueden usar como herramienta docente, comercial o para implementar votaciones en las grandes juntas de accionistas, programas de televisión o lugares de entretenimiento (cines interactivos, parques de atracciones). Quizás la más conocida es Replay Systems[3]. Estos sistemas están algo más próximos a la idea de democracia directa.

El sistema más cercano que conozco a la democracia directa, que he introducido anteriormente, es *freevote*, un sistema de voto electrónico *online* usado en HispaLinux. Básicamente *freevote* es un software distribuido bajo licencia libre, que permite realizar votaciones electrónicas en la red y que por tanto permite ejercer directamente el control de una organización o sociedad a sus miembros. Por supuesto, se trata de un sistema que todavía está muy verde, que puede ser cuestionado desde múltiples aspectos (seguridad, trazabilidad, etc.), pero es una muestra de que es posible implementar este tipo de sistemas.

La oposición a la implantación de este tipo de votaciones no tiene fundamento técnico. De hecho, existe una crítica muy feroz a la democracia directa como sistema sobre todo desde la casta de los políticos. Desde su punto de vista, un sistema político basado en estos principios estaría abocado al *populismo*, lo cual por cierto no deja en muy buen lugar sus actuaciones, pues quiere decir que ellos no son realmente los representantes del pueblo. Otra crítica es la propensión de estos sistemas a la demagogia.

De nuevo el mundo del software libre está indicando cuál puede ser el camino para resolver estos problemas. Por ejemplo, llevamos bastante tiempo usando sistemas de filtrado y calificación de las argumentaciones, que se usan con relativo éxito en los *weblogs* como BarraPunto. Con sistemas como éste, es la propia comunidad la que va evaluando las opiniones, permitiendo a los lectores circunstanciales (o con poco tiempo) leer solamente los aspectos más interesantes. Al ser evaluado por la comunidad, este sistema reconoce los argumentos de autoridad, porque tiene en consideración a los autores y porque requiere participar para poder opinar.

Como de costumbre, para muchos eso de la democracia directa puede sonar a utópico o irrealizable. Yo creo que es un sistema más justo y que a día de hoy podría ser implementable. Quizás fuese el momento de empezar a pensar en él, probándolo por ejemplo en algunos niveles de la administración. Quedan por supuesto muchos aspectos por resolver: la redacción técnica de las leyes, la situación de los funcionarios, de la justicia, etc.

---

[2] `http://www.indra.es`

[3] `http://www.replysystems.com/`

## Algunos enlaces

Para terminar acompaño el artículo con algunos enlaces más que permitan a los lectores crearse su propia opinión, indicando sistemas que ya funcionan, definiciones y críticas a estas ideas:

- Sistema libre de voto de electrónico (Freevote) usado en HispaLinux:
  `http://oasis.dit.upm.es/~jantonio/`
- Definición de *tecnodemocracia* en Wikipedia:
  `http://www.wikipedia.org/wiki/Techno-democracy`
- Discusión sobre *democracia directa* en Wikipedia:
  `http://www.wikipedia.org/wiki/Democracy`
- Argumentación contraria al uso de sistemas electrónicos de Rebecca Mercuri:
  `http://www.notablesoftware.com/evote.html`

# El Diccionario de la Real Academia de la Lengua

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

*Habemus* nuevo diccionario. O más bien, *nuevo Diccionario*, dado que no hablamos de uno cualquiera, sino del que elabora la Real Academia de la Lengua Española (RAE), el DRAE. Sin embargo, sólo es nuevo porque incluye nuevos términos y actualizaciones. En lo que toca a su distribución, a su estructura y a su modelo de elaboración sigue siendo el mismo diccionario de los últimos siglos.

Por ejemplo, parece poco razonable que la que debería ser la principal herramienta de trabajo con nuestro idioma se distribuya en unas condiciones que la ponen fuera del alcance de cualquier hablante. Por otro lado, en este momento de la historia, cuando por fin tenemos herramientas que permiten la comunicación fluida entre decenas de miles de personas, el DRAE sigue realizándose mediante un proceso relativamente cerrado, y su producto es una obra estática que no refleja ni siquiera una parte mínima de las posturas con respecto al idioma, ni su constante cambio, ni su enorme diversidad geográfica, ni sus diferentes usos según comunidades (aunque ha hecho grandes esfuerzos en estas direcciones).

¿No habrá llegado ya la hora de que la RAE pruebe nuevas formas de distribuir sus compilaciones? ¿No será el momento de plantearse nuevos desarrollos del concepto de diccionario? ¿No se podrá dar cabida en su elaboración a muchos más especialistas, e incluso a otros hablantes interesados, aunque su especialidad principal no sea el estudio de nuestro idioma?

## El nuevo diccionario de la RAE

Hace pocas semanas, con gran fanfarria, se celebró en Valladolid (España) una reunión de eruditos interesados en el idioma español. En ella se presentó el nuevo DRAE, que para muchos representa la norma del idioma, la *corrección lingüística* por excelencia. No son pocos los que ven en él también la herramienta indispensable para trabajar con el idioma. Debería, por lo tanto, estar al alcance del mayor número posible de hablantes, y en particular de todos aquéllos cuya herramienta de trabajo es precisamente el idioma. Por otro lado, la Real Academia de la Lengua, que es la institución que compila esta obra, parece estar interesada en la promoción y difusión del idioma español, y en ayudar en la medida de sus fuerzas a su uso *correcto*. La RAE parece también estar al tanto de que corren nuevos tiempos, y que tecnologías como Internet están cambiando las formas tradicionales de difusión del conocimiento.

Sin embargo, el diccionario de la RAE se comercializa en los mismos términos que hace unos cuantos siglos. Si quieres tener acceso a él, tienes que comprarlo

en una librería (por un precio no muy asequible para las economías de muchos hablantes), en su versión de papel o, en un alarde de modernidad, en un CD-ROM que no funciona más que en ordenadores equipados con software de determinada marca. O también, colmo de los colmos, puedes realizar consultas por Internet de una forma que parece especialmente diseñada para desincentivar su uso. Y *por supuesto*, todo él está cuidadosamente *protegido* de forma que su copia y redistribución está completamente prohibida.

¿Es ésta la mejor forma de promover el uso correcto del español? ¿Es lo más eficiente que se puede lograr (en términos económicos y sociales) para poner esta obra fundamental a disposición de los hablantes?

En cuanto a su factura, el diccionario se sigue elaborando con el mismo proceso que se usaba en el siglo XVIII. Se han incorporado mejoras técnicas que lo aceleran y que mejoran la calidad del resultado, pero sigue siendo en lo fundamental un proceso relativamente cerrado (en él participa una parte ínfima de los estudiosos del idioma, y no digamos de los hablantes) que produce una obra monolítica y *estática*. La informática ha entrado en la Academia, pero se ha usado sólo para acelerar el proceso de elaboración y para evolucionar del medio papel al medio digital. No se ha aprovechado para pensar de nuevo la forma de elaboración de un diccionario, ni el propio concepto de diccionario en sí. Sencillamente, sin reflexión, se sigue dando por bueno el método y el concepto que se desarrolló hace varios siglos, cuando construir un diccionario era algo novedoso.

¿Realmente no hemos avanzado nada en todo este tiempo? ¿No tenemos nuevas posibilidades para difundir el conocimiento, sino que seguimos relegados a seguir los dictados del pasado?

Naturalmente, estos razonamientos y estas preguntas se pueden aplicar a cualquier diccionario, y a muchas otras obras de difusión del conocimiento. Pero en el caso del DRAE son de especial aplicación. Por un lado, porque a la Academia se le supone más interés en la difusión y promoción del conocimiento relativo a la lengua que en la obtención de beneficios meramente económicos. Por otro, porque debería estar en la vanguardia de la investigación, como lo estuvieron las Academias de la Lengua de muchos idiomas en su momento.

## Un nuevo modelo de distribución

Desde que tenemos diccionarios, la razón fundamental de su existencia es su difusión, tan masiva como permita la tecnología de la época. Fue la imprenta la que hizo posible la edición de obras de gran tirada. Al amparo de esta tecnología nacieron las Academias de la Lengua y otros grupos interesados en compilar diccionarios que pudieran servir a la vez como herramienta de difusión del idioma y como ayuda a la comunicación entre los hablantes.

Desde hace al menos dos siglos se ha usado habitualmente un modelo de distribución para los diccionarios que ha sido muy ventajoso para todas las parte implicadas. Los autores del trabajo ceden, total o parcialmente, los derechos de

comercialización a una editorial. A cambio de esta cesión, la editorial se encarga de darle la mayor difusión posible, consiguiendo los recursos necesarios de la venta de ejemplares. Éste es, en general, el modelo clásico de difusión de obras escritas.

Sin embargo, también desde hace tiempo, conocemos otros modelos aplicables especialmente cuando la meta del trabajo no es obtener el máximo beneficio posible de la venta de ejemplares, sino difundir algún tipo de conocimiento. Así, por ejemplo, muchas instituciones financian la elaboración de trabajos o recopilaciones que luego son regaladas, o vendidas a bajo precio a las comunidades que pueden estar interesadas en ellas. No sería exagerado pensar que éste podría ser un buen modelo para la difusión de un diccionario. Tampoco que podría haber grupos interesados en fórmulas de patrocinio que permitiesen que el precio de cada ejemplar del DRAE disminuyese, probablemente, hasta llegar a cero.

Afortunadamente vivimos en un mundo que nos permite explorar más posibilidades en esta dirección. Para llegar a ellas, analicemos cuáles son los costes que conlleva el acercar un diccionario a cualquier hablante. Estos son, fundamentalmente, los siguientes: coste de elaboración del propio contenido del diccionario, coste de edición (en papel o en CD-ROM), coste de distribución y coste de publicidad y promoción.

Si nos olvidamos por un momento de los costes de elaboración del contenido (que trataremos más adelante), nos quedan los costes relacionados, directa o indirectamente, con la colocación del diccionario en casa de sus usuarios. Pero afortunadamente las cosas han cambiado mucho en este campo desde los tiempos de Gutemberg. Hoy día somos capaces de colocar un contenido en cualquier parte del mundo, en grandes cantidades, a coste prácticamente cero. Puede hacerse mediante un método bien probado y usado ampliamente en, al menos, una gran comunidad: la del software libre.

Este método consiste en una cuidadosa combinación de licencia de redistribución y tecnología. Por el lado de la tecnología, basta con conseguir una versión electrónica de buena calidad. Y no estoy refiriéndome a un buen programa de consulta del diccionario, que eso vendrá solo, sino de una buena estructuración de la información que lo compone. Lenguajes de descripción de contenidos, como los basados en SGML y XML, serían de gran ayuda. Por el lado de la licencia, habría que diseñar una que permita copiar y redistribuir el diccionario fácilmente (al estilo del software libre), asegurando su integridad cuando sea preciso, y permitiendo mejoras.

Si se atacan adecuadamente estos dos frentes, acabaremos con un diccionario en un formato para el que es fácil construir herramientas de consulta (que podrían realizar terceras partes, ahorrando de esa forma el coste de desarrollo), y que tiene un canal de distribución asegurado con coste cero (todas aquellas empresas interesadas en distribuir copias del diccionario a bajo precio). La Academia se podría dedicar a su verdadero trabajo, a lo que mejor sabe hacer: compilar el

diccionario. Y el resto, la comercialización, quedaría en manos de la sociedad y las empresas, que en estas condiciones lo harían con entusiasmo.

## Un nuevo modelo de desarrollo

Actualmente la elaboración del DRAE se basa en el trabajo de un número relativamente reducido de expertos, trabajando intensamente en los nuevos términos, y en las actualizaciones de los demás. Con este trabajo se compone, cada tantos años, una nueva edición.

Si buscásemos una analogía con el desarrollo de software, este sería un modelo tradicional de software propietario, el que Eric Raymond denomina *desarrollo estilo catedral*. El nombre es ilustrativo de su funcionamiento: el software (o en nuestro caso, el diccionario), se construye como se construían las catedrales. Un grupo reducido de gente (a veces una sola persona) es responsable del diseño completo de la obra, y dirige a otros grupos que se dedican a partes específicas del desarrollo. A su vez, esos grupos pueden diseñar los detalles de las partes a su cargo, y cuentan con equipos que trabajarán en la construcción de esos detalles. El resultado es un modelo jerárquico, muy probado y conocido, y especialmente útil cuando hay que organizar un grupo relativamente pequeño de gente, bien especializada en sus respectivos campos y bien disciplinada a la hora de trabajar. Podríamos decir, más formalmente, que se usa una planificación centralizada.

Pero hace tiempo que sabemos que hay otros modelos de desarrollo posibles. Por ejemplo, el mismo Eric Raymond identifica el *modelo de bazar*. De nuevo, el nombre es ilustrativo: el desarrollo sigue principios similares a los que rigen el funcionamiento de un bazar, un mercadillo o, en general, un mercado libre. No hay ninguna autoridad central que especifique qué productos, ni en qué cantidad, han de venderse en el bazar. Sin embargo, los compradores acuden a él porque encuentran lo que buscan, y los vendedores pueden mantener sus negocios porque tienen clientes dispuestos a adquirir sus productos. Cada comprador compra según sus preferencias, y cada vendedor decide por su cuenta qué productos ofrece, y a qué precios. Naturalmente, cada actor no puede hacer exactamente lo que quiere, sino que todos están sometidos a las reglas de funcionamiento del bazar (básicamente, las leyes de oferta y demanda). La planificación del sistema es completamente distribuida.

Por supuesto, es necesario estudiar con detalle cómo se puede estructurar el esfuerzo de creación o actualización de un diccionario con un modelo estilo bazar. Fueron necesarios varios años para descubrir y depurar las formas de gestionar proyectos libres que se usan hoy día, y sólo después de mucho ensayo y error se empiezan a vislumbrar las reglas que maximizan las probabilidades de éxito. En el caso de los diccionarios, ese proceso prácticamente no ha comenzado aún. Pero hay algunas ideas que podrían experimentarse. Por ejemplo, pueden usarse foros y sistemas similares a los de control de errores en programas para conseguir la colaboración coordinada de un gran número de expertos. Pueden usarse técnicas

clásicas de división en subproyectos para conseguir actualización en léxicos particulares para diversas comunidades. Las técnicas de certificación y cadenas de confianza pueden servir para dar más valor a las mejores contribuciones. Las tecnologías de la información en general pueden ser útiles para facilitar la recepción de colaboraciones y la revisión sistemática de todos los trabajos. Y, por supuesto, todo el proceso de elaboración puede ser cuidadosamente informatizado. De hecho, la construcción de las herramientas necesarias para hacer posibles estas ideas podrían ser en sí mismas un buen proyecto de software libre.

El software libre ha demostrado ya que los modelos estilo bazar pueden funcionar para producir software, si se dan las condiciones adecuadas. ¿Por qué no podemos experimentar con este tipo de modelos para producir un diccionario de calidad? Desde luego, producir un buen diccionario es una tarea enormemente compleja y especializada. Pero también lo es producir un sistema operativo completo. Antes de que se produjese el primer diccionario, cualquiera habría dicho que la tarea era inabordable. Sólo su finalización demostró, por el contrario, que era un trabajo abordable con unos recursos limitados. Igualmente, sólo el ponerse manos a la obra asegurará, quizás algún día, que se pueden construir diccionarios con un *modelo estilo bazar*. ¿Veremos algún día ese día?

## Hacia un diccionario libre

La Real Academia de la Lengua está, por muchos motivos, en una posición inmejorable para liderar este avance hacia un nuevo modelo de diccionario. Sin abandonar los mecanismos tradicionales, y con un coste muy bajo, podría reunir a un grupo interesado en experimentar en estas direcciones. Si el trabajo se hace bien, y las ideas de fondo no están equivocadas, en poco tiempo deberíamos comenzar a obtener resultados. Quizás un pequeño diccionario especializado en alguna materia, quizás una obra limitada a términos nuevos. Con el tiempo se podría definir mejor el proceso de desarrollo, depurando y completando las herramientas informáticas, afinando los mecanismos para permitir colaboraciones voluntarias y para identificar las más interesantes. Y abordando la tarea de lograr, con ayuda de muchos, un diccionario libre.

¿Se atreverá la Academia a moverse en esta dirección? Sólo el tiempo lo dirá. Pero si ellos no lo hacen, y en el más puro espíritu del software libre, quizás otro grupo en una posición no tan buena lo intente. Si lo logra, puede que dentro de unos años estemos preguntándonos para qué queremos una Real Academia de la Lengua.

## Aclaración final

Sé que la RAE elabora muchas más obras que su Diccionario, y que sus funciones no son únicamente compilarlo. Pero mucho de lo dicho aquí es aplicable

también a otras de sus producciones (por ejemplo, su corpus lingüístico del español, o su gramática). Y en cualquier caso, el DRAE es, sin duda, su obra más visible. Por ello este artículo se ha centrado en él.

# La historia cercana

Desde que Richard Stallman abandonó el MIT (Instituto Tecnológico de Massachusetts) para iniciar el movimiento del software libre ya han pasado 20 años. Entonces una impresora de última generación traía de cabeza a Stallman (y a todo el equipo de inteligencia artificial del MIT), ya que a la hora de imprimir se *atragantaba* frecuentemente con el papel. Cuentan que era decepcionante llegar al cuarto de la impresora y ver cómo tu trabajo todavía no estaba impreso, porque se había quedado atascada en uno anterior. Stallman decidió llamar a la empresa que fabricaba la impresora y les pidió el código fuente del driver de la impresora para añadir una funcionalidad en principio bastante simple para un buen programador como era él: que cuando la impresora se atascara, avisara a la persona cuyo trabajo estaba siendo imprimido para que éste la desatascara de inmediato.

Lo que recibió por respuesta fue una negativa, ya que el código fuente de esa aplicación era parte sustancial de la propiedad intelectual y de la ventaja competitiva de la empresa fabricante. A Stallman parece que esto no le sentó muy bien y seguidamente decidió lanzar un manifiesto en favor del software libre y crear un proyecto cuya finalidad era tener un entorno informático completo que usara únicamente software libre. Y el resto es historia...

Y de toda esa historia, la más cercana, la relativa a los últimos años es la que se comenta en la siguiente serie de artículos. Así, *¿Cómo van los proyectos de software libre?* es un breve resumen de lo acontecido en el año 2000 dentro del panorama del software libre. En esos tiempos, los grandes logros fueron pasar del entorno bidimensional de la shell (de uso exclusivo sólo para hackers, al menos todavía no he conseguido que mi madre haga uso de ella) a uno tridimensional, algo comúnmente conocido como la *conquista del escritorio*.

El ensayo *¿Está GNU/Linux listo para su uso masivo?* aborda el estado de GNU/Linux en la segunda parte del año 2001, cuestionándose si pasarse al sistemas GNU/Linux era una tarea abordable para el usuario de a pie. Ya entonces, a pesar de los más que notables avances que hacían que el uso de software libre posiblemente pudiera ser utilizado por casi todo el mundo, muchos -incluso *linuxeros*- todavía pensaban que el software libre no estaba lo suficientemente maduro. Como el avispado lector habrá notado ya, situar este artículo en el capítulo dedicado a comentarios históricos no es más que fiel reflejo de que el debate a día de hoy posiblemente esté más que cerrado: el software libre está listo

para el uso general. O, al menos, eso piensa el editor de esta obra, quién sabe si llevado en demasía por el optimismo.

En el análisis de lo que pasó en el 2001, que se podrá encontrar en *Y pasó otro año*, se puede ver cómo el software libre había dejado ya atrás la idea de ser el sistema de unos pocos elegidos para ser utilizado de forma notoria, a veces incluso masiva, en cualquier entorno y para cualquier dispositivo. Este año también es recordado como uno de los más duros dentro del sector de las tecnologías de la información: los felices años de finales de milenio dieron pie a la crisis de las *punto com* y el sector empresarial ligado al software libre, todavía una solución por la que había que apostar fuerte, sufrió en sus propias carnes la falta de fondos y confianza.

El último repaso histórico corresponde a *Mis notas sobre el 2002*. Las estrellas, entre otras que también se mencionan, son las grandes apuestas de las administraciones que se apoyan en el uso de software libre (LinEx), la disponibilidad de soluciones fáciles ya no sólo de usar, sino también de instalar (Knoppix), la aparición de una *suite* ofimática completa (OpenOffice.org) y la consolidación de un navegador rápido, potente y completo (Mozilla) que ha dado pie a una gran familia de aplicaciones. Por último, también se comentan los *peligros* que acosan al software libre, un tema que será tratado convenientemente y en mayor profundidad en el siguiente capítulo.

# ¿Cómo van los proyectos de software libre?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux Número 2, pág. 12-13, Noviembre de 2000, derivado de una parte de la ponencia "Actualidad del software libre", presentada en el III Congreso de HispaLinux

El software libre no sería más que un modelo posible, pero vacío de contenido, si no fuera porque existen productos utilizables. Y detrás de cada uno de estos programas, hay un proyecto que lo desarrolla y lo mantiene. Son estos proyectos los que aseguran que los programas libres siguen mejorando técnicamente, los que aseguran su calidad y los que materializan la colaboración de las comunidades de usuarios. En algunos casos, los proyectos funcionan de forma relativamente informal, dirigidos por voluntarios. En otros, hay empresas colaborando con muchos recursos a que los proyectos continúen adelante. Y en otros, están directamente promovidos por alguna empresa. A continuación, pasamos revista a los más destacados.

## Linux sigue vivo y con buena salud

El proyecto Linux es posiblemente el proyecto libre más conocido. Como es bien conocido, fue iniciado por Linus Torvalds, un estudiante finlandés que, con al ayuda de cientos de programadores de todo el mundo, construyó a principios de los años 1990, desde cero un kernel de sistema operativo similar a Unix. Al portar a ese kernel muchas aplicaciones libres ya disponibles en esa época (y en especial las producidas por el proyecto GNU), vio la luz el sistema operativo Linux, o como muchos preferimos nombrarlo, GNU/Linux. Hoy día el proyecto sigue siendo coordinado por Linus Torvalds, aunque otros desarrolladores, como Alan Cox, tienen en él un papel cada vez más destacado. Ya desde hace unos años han decidido liberar series de versiones con más frecuencia para seguir más de cerca los nuevos desarrollos[1]. En cualquier caso, se va a continuar con la política de mantener dos series en paralelo, la estable (pensada para usuarios finales, actualmente la 2.2) y la inestable (pensada para desarrolladores que quieren probar las últimas características del kernel, actualmente la 2.3). Alguno de los mejores sitios con información sobre el kernel Linux son Linux.com[2] y Kernelnotes[3].

La gran novedad del proyecto Linux es la serie 2.4, ya en fase avanzada de pruebas. No hay muchos cambios radicales, pero incluye más y más características

[1] Una serie de versiones está compuesta por todas las versiones con características y diseños similares. Actualmente, la serie estable es la 2.2, que incluye por ejemplo a las versiones 2.2.1, 2.2.5, 2.2.8, etc.

[2] `http://www.linux.com`

[3] `http://www.kernelnotes.org/`

avanzadas. Varios subsistemas han sido rediseñados, mejorando su rendimiento y sus capacidades. Incluirá muchos más manejadores para diferentes tipos de hardware. Algunas características esperadas desde hace tiempo estarán disponibles por fin (por ejemplo, soporte completo para plug-and-play, USB, nuevos sistemas de ficheros, etc.). En general, la mayoría de estas mejoras buscan hacer Linux más apto para su empleo en ordenadores de sobremesa, que es por ahora el segmento en el que mostraba más problemas. Puede encontrarse información más detallada sobre Linux 2.4 en el artículo "Wonderful World of Linux 2.4"[4], de Joe Pranevich.

## GNU produce más y más software

El proyecto GNU[5] fue iniciado por Richard Stallman en 1984, con la idea de producir un sistema operativo completo libre. Comenzó construyendo, sobre todo, herramientas para programadores (el compilador GCC, el editor Emacs, el depurador GDB, y muchas más) y utilidades típicas de sistemas operativos. Cuando estas herramientas fueron portadas al kernel Linux a principios de los años 1990, permitieron el nacimiento del sistema operativo GNU/Linux. Desde entonces, el proyecto GNU no ha dejado de producir virtualmente cientos de programas libres. Actualmente ha desarrollado también un kernel (HURD), que ya es muy utilizable, y es la base, por ejemplo, para una distribución Debian. Muchos de sus programas están entre los mejores en su campo (por ejemplo, GCC es sin lugar a dudas el compilador que genera código máquina en más plataformas), y GNU se ha convertido en garantía de calidad. El proyecto más innovador desarrollado dentro del marco del proyecto GNU es GNOME, que tiene tanta entidad como para ser considerado un proyecto por sí mismo.

Y a pesar de esta enorme producción de software, desde muchos puntos de vista, la mayor contribución de GNU ha sido de otro tipo: sentar parte de las bases legales (con la licencia GPL, una de las más usadas por los programadores de software libre) y filosóficas (al menos parcialmente) del movimiento de software libre. Actualmente, GNU está embarcado en la promoción de una nueva licencia, ésta para proteger documentación, y con una filosofía muy similar a la GPL. También se está trabajando en una modernización de la licencia GPL, que incluya las nuevas modalidades de uso de software, como por ejemplo los servidores de aplicaciones en Internet. Cuando sea publicada, esta licencia será la GPL 3.0.

---

[4] http://linuxtoday.com/stories/10698.html

[5] http://www.gnu.org

## Apache domina su nicho

Apache es el servidor de web usado en más sitios de Internet (más del 60 % de los sitios web usan Apache, según la encuesta de Netcraft[6]), y desde el punto de vista técnico es, sin duda, uno de los más completos y estables. La nueva serie de Apache (la 1.3) está ya disponible para su uso. Su rediseño ha sido muy completo y ahora es más modular. También se ha cuidado especialmente el rendimiento y la configuración, que se ha mejorado sustancialmente y se ha cuidado mucho el soporte para Windows NT y Windows 95. Alrededor de Apache están floreciendo otros proyectos, como Jakarta (integración de Java y Java servlets con el servidor de Web), que hacen que se mantenga en el frente tecnológico en este mercado. En cuanto a la organización del proyecto, se ha constituido la Apache Software Foundation, a la que pertenecen los desarrolladores que más han contribuido a Apache, y que tiene entre sus misiones coordinar los esfuerzos realizados en torno a este programa, así como su desarrollo futuro.

## Mozilla empieza a dar resultados

El proyecto Mozilla fue iniciado por Netscape (hoy parte de America Online) como el primer proyecto de software libre de gran escala iniciado por una empresa. Los recursos puestos por Netscape han sido enormes, incluso para proyectos *tradicionales* (propietarios). Cientos de programadores, docenas de herramientas auxiliares, y sobre todo una gran apuesta. Durante mucho tiempo, Mozilla fue considerado como un fracaso, hasta que hace unos meses empezaron a ver la luz las primeras versiones beta del producto. Parece que Mozilla va a ser un navegador que nos va a mostrar cómo va a ser la nueva generación de navegadores. Y a su alrededor ya está apareciendo toda una constelación de nuevas aplicaciones que usan componentes suyos (como Gecko, su motor de HTML), o están derivados de su código (como ChatZilla, un cliente de IRC que ha sido desarrollado usando gran parte del código de Mozilla).

## Debian incluye 4500 paquetes

Debian es una distribución de GNU/Linux (el kernel Linux más cientos de programas a su alrededor) que tiene la peculiaridad de no estar directamente promovida por una empresa, sino por cientos de desarrolladores repartidos por todo el mundo. Fue una de las primeras distribuciones, comenzada por unas decenas de desarrolladores a mediados de los años 1990. Hoy día es un gran proyecto coordinado en el que trabajan con diversos niveles de dedicación cientos de desarrolladores. El trabajo de estos desarrolladores consiste fundamentalmente en empaquetar aplicaciones para su inclusión en la distribución y la creación de herramientas que simplifiquen la instalación y la administración del sistema.

---

[6] `http://www.netcraft.com/survey/`

La nueva versión (Debian 2.2, alias Potato) vio la luz en agosto[7] Incluye más de 4.500 paquetes diferentes, entre los que se puede encontrar casi cualquier programa libre disponible para GNU/Linux. Una de las características fundamentales de esta distribución es la facilidad con la que se pueden realizar las actualizaciones de forma prácticamente transparente usando CDs o directamente Internet. Pero lo que más diferencia a Debian de otras distribuciones es su énfasis en que todo el software de su distribución principal sea software libre. En esta línea, Debian está consiguiendo ser la distribución libre por excelencia, que incluye virtualmente el estado del arte en software libre.

## KDE tiene cientos de aplicaciones

KDE es un entorno completo de escritorio que incluye ya cientos de aplicaciones que funcionan de forma integrada, incluyendo herramientas de ofimática (KOffice, que integra procesador de texto, hoja de cálculo, navegador, etc), de programación (KDevelop, un entorno integrado para la programación en C y C++), etc. KDE funciona en sistemas tipo Unix, y entre ellos en GNU/Linux. La versión 2.0 de KDE ha sido publicada en octubre[8]. KDE se incluye en muchas distribuciones de GNU/Linux como el entorno de escritorio por defecto, hasta el punto que muchos usuarios están ya identificando la apariencia habitual de KDE con GNU/Linux. Alrededor de KDE están surgiendo también empresas cuyo modelo de negocio está basado en el desarrollo integración y mantenimiento de aplicaciones dentro de este entorno.

## GNOME avanza con su modelo de componentes

GNOME es otro entorno de escritorio que también incluye varios cientos de aplicaciones. Su característica principal es su énfasis en un diseño arquitectural completamente basado en componentes que usan CORBA para integrarse y coordinarse entre ellos. Es notable cómo algunas empresas, principalmente HelixCode y Eazel están haciendo desarrollos basados en GNOME y contribuyendo muy activamente a su desarrollo. Algunas aplicaciones muy interesantes que están apareciendo en el marco de este proyecto son: Evolution (un programa para trabajo en grupo), Gnumeric (una hoja de cálculo), Nautilus (un gestor de ficheros), GIMP (un programa de tratamiento de imágenes), y AbiWord (un procesador de textos). La constitución de la Fundación GNOME, que tuvo lugar este verano, y en la que participan empresas como Hewlet Packard, SUN, IBM, Eazel y HelixCode demuestra el gran interés de la industria informática por este proyecto.

---

[7] Nota del editor: El autor se refiere a agosto de 2000.

[8] N. del E.: El autor se refiere a octubre de 2000.

## XFree86 ya ha liberado la versión 4.0

XFree86 es la implementación de X Window que utilizan casi todos los sistemas operativos libres sobre procesadores derivados del i386. Proporciona la infraestructura sobre la que están construidas la mayoría de las aplicaciones gráficas libres. Acaba de liberar una nueva versión, la 4.0, que aún no es completamente estable, pero ya muestra los beneficios de un rediseño casi completo. El camino hacia esta nueva versión comenzó hace casi dos años, y ha conseguido una mayor modularización del sistema, mejoras en el rendimiento y nuevos servicios.

## Hacia dónde vamos

Parece que -en contra de muchas previsiones pesimistas- los proyectos de software libre siguen vivos y coleando. Más allá de personalismos y de divisiones, continúan produciendo software de buena calidad. Y continúan demostrando cómo se pueden gestionar proyectos con cientos de desarrolladores repartidos por todo el mundo, y aún así mantener la coherencia necesaria para que las publicaciones tengan lugar a buen ritmo. El futuro dirá si todo esto puede mantenerse y mejorarse, pero por ahora parece que se progresa a buen ritmo.

# Y pasó otro año

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 2, pág. 12-13, Enero de 2002

Se nos acaba el 2001. Mirando hacia atrás, mucho ha ocurrido este año en el mundo del software libre. En el lado positivo, es de destacar el aumento sostenido del uso y del conocimiento del software libre, se mida por el parámetro que se mida. Entre lo negativo, es sin duda preocupante la gran cantidad de empresas relacionadas con este mundo que han cerrado o están teniendo muchas dificultades. Aprovechando el recuento típico de fin de año, voy a comentar sobre lo que me ha parecido más relevante de éste que termina.

## La conquista de los usuarios *normales*

El software libre en general, y GNU/Linux en particular, han seguido progresando en la larga marcha hacia la conquista de los ordenadores de los usuarios *normales*. Poco a poco GNU/Linux va entrando en empresas, normalmente de la mano de juegos de ofimática como StarOffice (o su versión libre OpenOffice.org), y en gran medida gracias a la facilidad de uso que proporcionan los entornos de escritorio GNOME o KDE. A esto también han ayudado, sin duda, las últimas mejoras en sencillez de instalación que han introducido casi todas las distribuciones.

Sólo cuando los usuarios puedan hacer tranquilamente en GNU/Linux todo lo que suelen hacer en otros sistemas esta marcha se acercará a su fin. En entornos como la ofimática o la programación esto está cerca de ser cierto. Sin embargo en otros, como los juegos, el camino es aún largo. Por eso hoy día incluso los *linuxeros* más acérrimos suelen seguir prefiriendo el arranque dual, y no se acaban de atrever a borrar *la otra* partición.

Pero además de necesitar más y mejores programas, hay otro problema que frena el avance de GNU/Linux entre los usuarios no técnicos, y que lo va a frenar más en el futuro si no lo evitamos. Es la imposición de uso de un determinado programa que se hace en muchos ámbitos. Es, por ejemplo, bien conocido que el censo español no puede rellenarse desde Internet más que con cierto producto de cierta empresa. Lo que lleva a la situación bastante curiosa de que se esté usando el dinero de los impuestos de los contribuyentes españoles para fomentar la posición de monopolio de una empresa estadounidense.

La única forma de denunciar y frenar estas situaciones es organizándonos y haciendo oír nuestra voz. Sólo unas asociaciones fuertes y reivindicativas, arropadas por una buena cantidad de miembros, podrán hacer frente con ciertas posibilidades de éxito a estas situaciones, mezcla de ignorancia e incompetencia, donde la libertad de elección que deberíamos tener se ve tan coartada.

Afortunadamente, ya estamos en ese camino. Campañas como la del PADRE libre (petición para que el programa que se utiliza para hacer la declaración del impuesto español sobre la renta esté disponible para plataformas como GNU/Linux) están empezando a dar sus frutos. En casi todos los países del ámbito hispano las asociaciones de usuarios de GNU/Linux o de software libre florecen, y en general gozan de muy buena salud. Aunque sólo el futuro dirá si son capaces de mantener una tasa sana de crecimiento, y si sus acciones estarán al nivel necesario como para invertir la actual tendencia que perjudica a las soluciones libres.

## Las empresas, una de cal y otra de arena

La crisis en el sector de Internet se ha instalado entre nosotros. Y a la sombra de ella, el capital riesgo ha huido de todo lo que suena a negocio "demasiado nuevo". Esto ha perjudicado a muchas empresas relacionadas con el software libre que estaban en medio de alguna ronda de financiación. Así, empresas tan prometedoras como Eazel han quebrado. Otras, tan emblemáticas como VA Linux, están capeando el temporal como pueden, en muchos casos con manifiesto nerviosismo y pérdida de rumbo.

Sin embargo, en estos mismos meses hemos visto la entrada decidida de grandes empresas informáticas en el mundo del software libre. La estrategia de IBM al respecto es manifiestamente clara, hasta el punto de haberse convertido en el mayor promotor del software libre en el mundo empresarial, con lo que eso supone. Otros gigantes, como HP-Compaq, están cada vez más cerca de este tipo de estrategias, quizás como única forma de competir.

Esta situación esquizofrénica (por una lado parece que cada vez más empresas *nuevas* atraviesan dificultades, por otro cada vez empresas *tradicionales* se convierten al software libre) es difícil de analizar. ¿Estamos más cerca o más lejos de que las soluciones libres se conviertan en las preferidas por el mundo empresarial? Parece que está aumentando la desconfianza en los modelos de negocio que se han propuesto hasta ahora en el mundo del software libre. Pero a la vez, parece que a muchas empresas no les queda más opción que entrar de cabeza en este mundo para poder ser competitivas...

En cualquier caso, lo que parece claro es que aún no se ha descubierto el modelo de negocio que permita generar recursos con solidez. Aún así algunas empresas, como Red Hat Linux o Ximian, podrían estar cerca de encontrarlo.

## La explosión de los proyectos

Donde no se ha notado ningún tipo de crisis es en la cantidad y calidad de los proyectos de software libre. Diariamente tenemos anuncios de nuevos proyectos, de nuevas liberaciones, y de nuevos campos donde se están construyendo programas libres. En este último año se percibe cómo la masa crítica de desarrolladores

se ha sobrepasado en muchos ámbitos que hasta ahora estaban relativamente desatendidos por el software libre (como los reproductores multimedia, por ejemplo). Y cómo aparecen las primeras incursiones exitosas en entornos casi prohibidos, como los ordenadores de mano (PDA).

El futuro del software libre depende de que siga siendo competitivo en los campos en los que lleva establecido desde hace tiempo (desde los compiladores y los entornos de programación hasta los entornos de escritorio y los programas de ofimática). Pero también de que sea capaz de colonizar nuevos territorios al menos con la misma penetración que el software propietario, y si es posible, un poco por delante de él. Por eso son muy importantes los nuevos campos que se han abierto, o se han empezado a consolidar durante este último año.

## Los medios

Este año ha sido el del establecimiento del software libre en el kiosco. Por un lado, las revistas dedicadas al software libre, o al menos a sistemas como GNU/Linux, han madurado y se han multiplicado. Ya no es raro encontrar en cualquier establecimiento publicaciones si no especializadas, al menos que incluyan en cantidad significativa información sobre software libre. Y en general, toda la prensa informática muestra secciones fijas sobre sistemas libres, y el tratamiento de temas relacionados con GNU/Linux ya ha dejado de ser lo excepcional para pasar a ser la norma.

Incluso la prensa generalista se encarga de vez en cuando de los avances del software libre. Y GNU/Linux se ha convertido en referencia obligada como única alternativa al monopolio de Microsoft. Sin embargo, esta misma atención ha traído sus problemas. En particular, el software libre está cada vez más bajo el radar de las grandes empresas de software propietario, que ya han puesto a funcionar su gran maquinaria de márketing para *combatirle*. En el tiempo que viene, cada vez contemplaremos ataques más directos, para los que habrá que estar preparados, y a los que habrá que responder de alguna manera.

## La situación legal

Lo más preocupante durante este año que termina ha sido el empeoramiento en el frente legal. Por un lado, Europa está más cerca que nunca de aceptar las patentes de software, a pesar de los esfuerzos para explicar cómo este cambio nos perjudicará a todos. Por otro, las legislaciones que tratan de defender los derechos de autor están siendo cada vez más incompatibles con el desarrollo de software libre. Especialmente preocupante está siendo el caso del DeCSS, que aún no se ha resuelto completamente (aunque en las últimas semanas parece estar reconduciéndose), y las legislaciones propuestas como sucesoras de la DMCA, que ponen directamente fuera de la ley muchos desarrollos libres, a la vez que limitan libertades fundamentales que siempre hemos tenido como consumidores.

Sin embargo, también ha sido este año el primero en el que se ha percibido un cambio de tendencia en el mundo del software libre en cuanto a la preocupación general por los asuntos legales. Por fin ha dejado de ser extraño ver desarrolladores de software libre preocupados por las implicaciones para ellos y para su trabajo de las leyes que se están proponiendo. Se percibe también, diría que por primera vez, una preocupación cada vez mayor por temas como la libertad de innovación o la libertad de elección de plataforma software y sus implicaciones sociales y políticas.

## De todo un poco

Resumiendo, el año en una frase, podría decirse que "Hemos tenido de todo un poco". En líneas generales, ha sido un año de avance, y esto es especialmente importante cuando ya hay varios sectores que han identificado al software libre como un enemigo a batir. Creo que es muy positivo no haber retrocedido cuando el entorno ha pasado de ser completamente indiferente a ser marcadamente hostil en algunos casos muy importantes. Y desde muchos puntos de vista no sólo el software libre no ha retrocedido en estas circunstancias, sino que ha continuado su carrera hacia adelante.

¿Podremos decir lo mismo el año que viene?

# ¿Está GNU/Linux listo para su uso masivo?

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 12, pág. 12-13, Septiembre de 2001

GNU/Linux en particular, y el software libre en general, están ante una encrucijada: permanecer como el juguete de una minoría, o pasar a ser *lo que usa todo el mundo*. De lo que ocurra en los próximos años va a depender el camino que se tome. Pero, ¿está listo el mundo GNU/Linux para avanzar por el camino del uso masivo? ¿Estamos listos los que creemos que el software libre ofrece muchas ventajas y un modelo de desarrollo viable para promover su uso en todos los entornos? Dicho más clarito, ¿nos creemos de verdad que el software libre puede llegar, en un plazo corto, a ser la solución preferida en muchos situaciones?

Aunque pueda parecer extraño, a mi alrededor veo que *los que estamos en esto* parecemos no creérnoslo. Cuando enfrentamos a un *linuxero* de toda la vida con la instalación del ordenador de su hermano, o con la informatización de la empresa en la que trabaja, en demasiadas ocasiones opina que es mejor no usar software libre. Y no es que sea un advenedizo, alguien sin experiencia, o que no controle mucho los últimos avances del mundo GNU/Linux. Es, simplemente, que piensa que todo esto aún no está listo para el usuario medio, ése que sólo quiere usar su ordenador. Que hacen falta muchos conocimientos técnicos para poder sobrevivir en el mundo del software libre, y que las cosas no son suficientemente sencillas.

¿Tenemos complejo de inferioridad, nunca nos hemos acabado de creer lo que hablamos a todas horas, o qué está pasando? Si te interesa mi opinión, sigue leyendo...

## Ni nosotros mismos...

Seguro que has visto más de una vez cómo un usuario veterano de GNU/Linux defiende vehementemente que aunque GNU/Linux es una gran cosa, y a él le va bien, aún no está listo para su uso por todo el mundo. Por ejemplo, hace unos días, tomando una cañas con varios *linuxeros* convencidos desde hace tiempo, y después de hablar sobre las últimas mejoras a los sistemas de ficheros transaccionales en el kernel, sobre la evolución de Mozilla y sobre lo maduras que están las últimas versiones de KDE, salió el tema de siempre. En este caso, los problemas de uno de los presentes instalando no sé que cosa en el *ordenata* de su hermano que, *naturalmente*, tenía Windows como sistema operativo. "¿Y eso?" "Pues me pidió que le ayudara cuando se lo compró, y claro, como no usa el ordenador más que para escribir y navegar por la red, no iba a ponerle GNU/Linux, que mi hermano no sabe nada de ordenadores...".

Te suena, ¿eh?

Yo me he encontrado situaciones de este estilo por decenas. Tantas que acabas por considerarlas normales y cuando te encuentras en una similar, sin pensarlo, directamente tomas la misma decisión. Pero me pregunto, ¿es realmente ésta la única opción? O más bien, ¿es ésta siempre la opción más conveniente? Porque si de verdad el software libre es algo tan bueno como todos creemos, ¿por qué negárselo a la gente de nuestro entorno, sea o no entendida en ordenadores?

Luego quizás hay que replantearse el asunto..

## Seamos objetivos

Naturalmente, no se puede decir que GNU/Linux (o cualquier otro sistema operativo libre) es siempre la mejor solución para un usuario dado. Pero lo puede ser en muchos casos, incluso si ese usuario no tiene muchos conocimientos informáticos. Y de todas formas, lo mismo ocurre con cualquier otro sistema propietario.

Desde mi punto de vista, los casos más claros en los que GNU/Linux es una buena solución para un usuario no técnico se dan cuando se cumplen las condiciones siguientes:

- El usuario sólo suele hacer ciertas tareas con el ordenador, y esas tareas pueden realizarse bien con GNU/Linux. Una situación típica es la del ordenador para trabajo de oficina (procesador de textos, hoja de cálculo, correo electrónico y navegador de web). En este caso, tenemos juegos de herramientas más que adecuados. Además, estas herramientas (por ejemplo, las disponibles en los entornos GNOME o KDE) son sencillas de usar, y tienen interfaces similares a cualquier otra herramienta de su categoría.
- El usuario tiene acceso a personas con conocimientos específicos sobre GNU/Linux. En estos casos, será fácil conseguir ayuda si se presenta algún problema puntual, por ejemplo, en el momento de la instalación. Esas personas tampoco son, a estas alturas, algo extraño. Por ejemplo tú, incluso con tus conocimientos básicos, puedes ser una de ellas. Tus amigos y tus familiares te tienen a su alcance (si tú les dedicas algo de tiempo, claro). Con tu ayuda, el uso de GNU/Linux puede ser mucho más placentero... Sólo piensa en lo que has ayudado ya a tus amigos a instalar o resolver problemas de Microsoft Windows y el tiempo que le has dedicado a ello... En entornos profesionales (empresas y similares) este personal se puede contratar. Basta con fijarse, cuando se cubren las plazas de informáticos en la empresa, en que sepan de GNU/Linux y software libre. O con ayudar al personal ya contratado a reciclarse, actualizando sus conocimientos con algún curso de GNU/Linux.
- El usuario puede beneficiarse directamente de alguna de las ventajas del software libre. Por ejemplo, si tienes conexión ADSL en casa, montar un buen cortafuegos con GNU/Linux, que puede actualizarse diariamente para evitar problemas de seguridad, puede ser un buen comienzo. Otro caso es el de

los usuarios a los que les gusta probar muchas aplicaciones. Qué mejor que regalarles las más de 2.000 que hay en los CDs de Debian 2.2...

Si piensas un poco en ello, seguro que se te ocurren más y más situaciones donde sí deberías recomendar, claramente, que al menos se instale un arranque dual, y quizás sólo GNU/Linux.

Como siempre, las cosas no suelen ser tan claras. También puede haber problemas, y habitualmente los hay. Pero normalmente, hay formas de solucionarlos:

- La instalación puede ser difícil. Hoy día, casi cualquier distribución de GNU/Linux es fácil de instalar, pero siempre resultará más difícil que no tener que hacerla... Y aún no es habitual que al usuario le venga GNU/Linux preinstalado. La única forma que yo conozco de resolver este problema es con ayuda. Ayuda humana (tener cerca a alguien que esté familiarizado con la instalación de GNU/Linux) y ayuda software (tener una instalación que cause los menos problemas posibles, y haga automáticamente la mayor parte del trabajo). Una cuidadosa selección de la distribución a instalar ayuda mucho en esta segunda parte.
- Problemas con el soporte para algún tipo de hardware. Desgraciadamente, mucho hardware (tarjetas gráficas, de sonido, de conexión, de red, etc) no tiene ningún tipo de soporte para GNU/Linux por parte de sus fabricantes. Todo lo que conseguimos cuando lo compramos es un CD con un driver para cierta versión de Windows. Por el lado bueno, sin embargo, tenemos la suerte de que la mayor parte de los dispositivos están soportados de forma nativa en Linux. Siempre hay problemas (el último modelo de tarjeta gráfica sólo está soportado por versiones experimentales del kernel, tal módem no tiene drivers para Linux, etc.). La única forma de evitar estos problemas es comprar el hardware con cuidado, si hace falta pidiendo garantía de que funcionará con GNU/Linux (lo que de paso envía al vendedor el mensaje de que si sabe de GNU/Linux tendrá un cliente más, y eso no es mala cosa). Si no se puede hacer, habrá que darse unas vueltas por Internet... Tenemos suerte de que GNU/Linux sea, con diferencia, el sistema sobre el que hay más información en Internet. Y de paso, si encuentras una solución, no te olvides de contribuir también. Documéntala y ponla en algún sitio en la Red. Si no sabes dónde, envíala a la lista de correo de un grupo de usuarios. Esas listas suelen archivarse, y más tarde alguien puede aprovecharse de tus sudores.
- No hay aplicaciones para alguna tarea que quiere realizar el usuario. Hoy día en GNU/Linux hay aplicaciones para casi cualquier tarea, y es conveniente echar un vistazo exhaustivo si parece que no existe lo que necesitamos: cada día aparecen nuevas cosas. Pero a veces, a pesar de todo, no encontramos algo. Por ejemplo, el último juego de matar no sé qué bichos. En estos casos, la única solución es mantener el arranque dual, y seguir mirando periódicamente por si apareciera lo que necesitamos. A veces WINE, que permite ejecutar aplicaciones Windows emuladas sobre GNU/Linux, puede ser también de ayuda.

- En algunas ocasiones simplemente ocurre que para el usuario es difícil conseguir una versión actualizada de una distribución de GNU/Linux, o decidir cuál es la que le conviene instalar. De vez en cuando alguien me cuenta sus problemas de instalación y después de charlar un rato me doy cuenta que está usando unos CDs de hace tres o cuatro años... Pero estamos de suerte, cada vez es más sencillo encontrar distribuciones recientes, bien bajándolas de Internet, comprándolas a distancia o directamente en alguna tienda de informática o grupo de usuarios. Si te interesa, aquí también puedes ayudar. Si tienes una distro actual y con permisos de redistribución *tal cual* (como por ejemplo, Debian), ofrece tu tostadora a tus amigos. Para las empresas, esto raramente es un problema. Con lo que están acostumbrados a pagar por licencias de programas propietarios, pagar lo que puede costar una buena distro no es ningún problema.
- Muchos usuarios echan de menos documentación actualizada para aprender a usar GNU/Linux. La mayoría de los libros que te encuentras en la zona de informática de las librerías no son más que manuales de aplicaciones sobre Windows, y es difícil encontrar verdaderos manuales sobre informática, y más aún sobre GNU/Linux en particular, especialmente si quieres la información en español. Por el lado positivo, ya empieza a haber libros *tradicionales* que se dedican a GNU/Linux, y siempre tenemos el almacén de información por excelencia: la Red. Sitios como LuCAS están llenos de información sobre GNU/Linux en español.

Como ves, estos problemas, aunque pueden ser frecuentes, no son insalvables. Aunque, por supuesto, hay más. ¿Son suficientes para desaconsejar la instalación de GNU/Linux en lugar de otros sistemas? Habría que hacer un estudio caso por caso, pero en general, creo que no. GNU/Linux puede ser una alternativa perfectamente válida para los usos de muchísimos usuarios *normales*, y si podemos, no está de más que les echemos una mano con sus problemas. Quizás con el tiempo vayan aprendiendo, y más adelante nos puedan ayudar ellos a nosotros...

## Algunas ideas

En el mundo del software libre no ocurre que la única opción sea esperar a que determinada empresa venga a resolver nuestros problemas. Aquí podemos ser parte activa de las soluciones. Si quieres ayudar a que más y más gente pueda usar GNU/Linux (u otro sistema libre) con provecho, hay muchas cosas que puedes hacer. Algunas ya las he ido contando, otras las puedes ver aquí, a modo de ejemplo:

- Apúntate a un grupo de usuarios, o funda uno nuevo con tus amigos *linuxeros* si no lo hay donde vives, estudias o trabajas. Los grupos de usuarios sirven estupendamente como redes de ayuda, donde los más veteranos pueden echar una mano a los más nuevos, y todos juntos pueden avanzar más rápidamente.

Pueden editar sus propios CDs con distribuciones, organizar sus jornadas donde se cuenten los últimos avances en GNU/Linux, discutir sobre problemas en listas de correo, etc.

- Si sabes otros idiomas, puedes ayudar traduciendo documentación al español. Mucha gente no tiene la opción de usar documentación en inglés, la necesitan en su idioma. Sitios como LuCAS ayudan a coordinar esta tarea de traducción, inmensamente útil y que nunca acaba.
- Documenta tus problemas (y tus soluciones). Si tienes más tiempo y crees que sabes escribir, procura preparar guías donde expliques lo que has aprendido sobre cualquier tema relacionado con GNU/Linux. Seguro que tu experiencia será muy útil a otros *linuxeros*. Además de LuCAS, hay otros sitios (como La Espiral[1]) donde puedes enviar lo que escribas.
- Si sabes programar, ayuda a mejorar los procedimientos de instalación y de administración de GNU/Linux. Colabora en alguno de los proyectos de software libre para automatizar estas tareas. Tu labor ayudará a reducir mucho la altura de las barreras de entrada para los que quieren aproximarse a GNU/Linux.
- Si estás en un entorno empresarial, evalúa las posibilidades de GNU/Linux en tu empresa, y habla con quien tome las decisiones al respecto. Procura que tu evaluación sea realista, pero no subestimes las posibilidades de GNU/Linux. Trata de identificar también las ventajas que obtendrían del uso de software libre, y propón un plan de introducción de Linux que pueda llevarse a cabo sin demasiados trastornos. Naturalmente, tu experiencia seguramente será muy valiosa si la empresa decide finalmente beneficiarse de GNU/Linux o de cualquier programa libre.

Y te apetezca o no hacer todo esto, al menos no hagas una cosa. Por favor, nunca, nunca digas que GNU/Linux no es de utilidad para un cierto usuario o para cierta tarea en una empresa si no estás absolutamente seguro de lo que dices. Muchas veces, lo que ocurre es simplemente que no sabes cómo GNU/Linux les puede ayudar, o de qué manera organizar el asunto para que el usuario salga beneficiado. No todos (ni siquiera los que saben mucho sobre software libre) conocemos todas las posibilidades ni tenemos todas las experiencias. Y ya hay demasiada gente que, con intereses claros porque quiere que uses otras opciones, se encarga de decir que GNU/Linux no sirve a los usuarios *normales*. Si no estás completamente seguro de lo que dices, y de que lo dices con suficiente conocimiento, por favor, no lo digas.

## Para terminar (o para empezar)

GNU/Linux está mucho más listo para su uso masivo que lo que mucha gente piensa. Ya hay muchas experiencias al respecto, y cada vez tenemos más.

---

[1] `http://www.laespiral.org`

Desgraciadamente, muchos ignoran esta situación, y siguen con la impresión de que el software libre es sólo para *entendidos*. Desgraciadamente, entre esta gente se encuentran también muchos *linuxeros* experimentados. Quizás sea hora de que nos replanteemos las posibilidades de GNU/Linux en cualquier entorno, y de que aprovechemos cualquier resquicio para ayudar. Por supuesto, las cosas hay que hacerlas bien, y si en un entorno concreto no vemos posibilidades, mejor no hacer propuestas que van a terminar en fracaso. Pero si después de considerar cuidadosamente las opciones entendemos que GNU/Linux tiene una posibilidad, pongamos de nuestra parte para que se convierta en realidad.

# Mis notas sobre el 2002

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 27, pág. 12-13, Diciembre de 2002

Esta noche nochevieja, mañana año nuevo. Y mi artículo sin escribir. Hoy todo el mundo haciendo balance... ¿sobre qué podría escribir yo? Claro, pues balance también. Así que como no se me ocurre nada mejor, si sigues leyendo, sabrás lo que más me llama la atención en este momento sobre lo que ha ocurrido durante 2002 en el mundo del software libre y afines (visto, naturalmente, a través de mi particular filtro).

## LinEx

Una de las cosas más interesantes del año ha sido, sin duda, el fenómeno LinEx. Mucho más allá de que tengamos una nueva distribución de GNU/Linux basada en Debian (que no deja de ser una anécdota), y más allá de su impacto en medios de comunicación, lo extraordinario es la (aparentemente) sólida apuesta de una administración pública por el software libre.

La Junta de Extremadura ha decidido probar un modelo diferente en la enseñanza de informática, y a medio plazo, probablemente en el propio uso de la informática dentro de sus competencias. Esto la convierte en la primera administración pública de un país desarrollado que toma este camino. Por ello en los próximos meses (y años) sus acciones y sus resultados serán escrutados con lupa desde muchos sitios. Alrededor de la iniciativa de la Junta ya está empezando a producirse movimiento, tanto dentro como fuera de Extremadura. Hay academias que enseñan informática con LinEx. Se han escrito libros para apoyar en esta enseñanza. Hay ordenadores que se venden con LinEx preinstalado. Y todo esto podría ser nada más que el principio.

Pero además de ofrecer muy buenas perspectivas, también hay muchos riesgos. Precisamente por estar bajo la lupa de todo el mundo cualquier equivocación, cualquier consecuencia indeseada, será amplificada y vendida en todo el mundo como un fallo del software libre.

## Knoppix

¿Qué te parece llevar en el bolsillo un CD que puedes poner en cualquier PC, para que al arrancar el trasto tengas un GNU/Linux completamente funcional? Eso es Knoppix. Une una (buena) detección automática de hardware con una buena selección de programas y un funcionamiento *en vivo* que no requiere una partición para instalarse, mételo todo en un CD, y listo. Indispensable para mostrar rápidamente *qué es eso de lo que siempre estás hablando*. O para cosas

más *serias*, claro, siempre que te venga bien ejecutar Linux en un PC que no lo tenga...

## OpenOffice.org

Creo que el anuncio de SUN de liberar OpenOffice.org bajo GPL no es de este año (ahora no estoy seguro) pero en cualquier caso, yo lo he descubierto como usuario en 2002. Y tengo que considerarme como un usuario muy, muy satisfecho. OpenOffice.org se ha convertido en un juego de aplicaciones ofimáticas de calidad al menos igual a la de cualquier otro juego que conozca. Además, salva aceptablemente bien la mayoría de los problemas de transferencia de datos con Microsoft Office. Por fin el software libre ha llegado, en condiciones tan buenas como cualquier otro, al mundo de la ofimática.

Por fin ya es posible cambiar, prácticamente sin traumas, de los entornos propietarios habituales en ofimática (sin duda la aplicación estrella en el mundo empresarial) a entornos completamente libres (por ejemplo, GNU/Linux más GNOME y/o KDE más OpenOffice.org). Todavía queda camino que andar por la vía de la integración entre aplicaciones, pero ya estamos ahí.

## Mozilla, Galeon y los demás

Mozilla, el proyecto de Netscape para construir un navegador libre, ya es una realidad. Durante bastante tiempo se habló del fracaso del proyecto. Hoy, como mucho, se habla del fracaso en cuota de mercado (tal y como están las cosas, Microsoft Internet Explorer es líder indiscutible, y no parece que la situación vaya a cambiar a corto plazo). Pero el proyecto ya ha dado sus frutos. Y en la mejor tradición del software libre, los resultados no han sido sólo los esperados (el navegador Mozilla), sino que además han surgido un montón de proyectos relacionados. Entre ellos, destaco Galeon sólo porque lo uso habitualmente, pero hay muchos más.

Mozilla ayuda a completar un hueco que teníamos en el mundo del software libre. Antes de la aparición de Konqueror, no había muchos navegadores libres con interfaz gráfica. Ahora, como mínimo tenemos por un lado a Konqueror, y por otro la familia de navegadores basados en Gecko, el visualizador embebido en Mozilla. Y además, ambos se desarrollan de forma saludable y a toda velocidad. La combinación Mozilla más OpenOffice.org permite usar software libre para las tareas más cotidianas incluso en un entorno Windows, lo que, por primera vez en la historia del software libre, permite otra transición simple: se puede empezar usando estas dos aplicaciones sin cambiar de sistema operativo, y con el tiempo eliminar la única pieza no libre pasando a GNU/Linux o FreeBSD.

## Debian 3.0

Para muchos, Debian 3.0 (Woody) no será más que una nueva versión de una distro basada en Linux. Pero para mí, usuario satisfecho de Debian desde hace tiempo, ha sido la oportunidad de poner al día mis utilidades, y sobre todo de contemplar cómo una vez más, lo inexplicable ha sucedido. Ver cómo cerca de 1.000 desarrolladores voluntarios, repartidos por todo el mundo y trabajando de forma coordinada, han conseguido poner juntos más de 4.000 paquetes fuente, con unos altos estándares de estabilidad y seguridad, es sin duda un privilegio. Desde luego hay otras distribuciones, y este año ha empezado su camino hacia la fama alguna con metas similares a las de Debian (ahí está Gentoo, por ejemplo). Pero para mí, Debian es Debian, y tengo que reconocer que me llena de orgullo el que mi trabajo haya ayudado, siquiera mínimamente, a que este proyecto siga adelante.

¿Y tú, aún no has instalado Debian 3.0? ;-)

## HispaLinux se acelera

HispaLinux lleva ya mucho tiempo promoviendo el software libre (hace unos meses celebró su V Congreso), pero yo diría que este año se ha acelerado mucho en esta tarea. Durante 2002, ha pasado de estar en cierta medida en un segundo plano a ser protagonista de muchas iniciativas cara a los medios, a las administraciones públicas, a las empresas y, por supuesto, a los interesados en el software libre. Los grupos de trabajo se han multiplicado, el número de socios ha crecido a toda velocidad, y las actividades en las que la asociación está implicada son cada vez más.

Desde luego, queda mucho camino por recorrer, y sin duda HispaLinux (como asociación, o alguno de sus miembros individualmente) ha cometido errores. Pero todo con todo, creo que es difícil no estar de acuerdo en que el balance final ha sido positivo, y el incremento de actividad (quizás un síntoma del incremento de actividad en el mundo del software libre en general) muy notable. De forma destacable, en mi opinión, hay que tener en cuenta el esfuerzo en dar cabida a las iniciativas de los socios, y el apoyo que en muchos casos recibe la gente que tiene ganas de hacer cosas. Aún queda mucho por hacer, mucho por mejorar, y muchas iniciativas por promover. Habrá que ver si HispaLinux sigue esta tendencia de crecimiento, o acaba estrangulada por su propio éxito. Pero creo que es seguro que en gran parte dependerá de lo que hagamos sus socios (incluyéndote a ti, si ya lo eres, o si lo quieres ser). Al fin y al cabo, no hay que olvidar que (al menos por ahora) todo el trabajo que se realiza en HispaLinux está basado en el esfuerzo de voluntarios...

## Ingeniería del software libre

Llevaba ya algunos años intentando estudiar el software libre desde diversos puntos de vista, y sus implicaciones relacionadas con la ingeniería del software habían sido durante ese tiempo uno de mis intereses. Pero ha sido durante este año cuando me he podido dedicar más a ello, y cuando he percibido un interés creciente al respecto. Los modelos de desarrollo que se usan en los proyectos de software libre son tan sorprendentes que, en muchos casos, sólo el hecho de que han funcionado permite creer que pueden funcionar. Pero en general, sabemos aún muy poco sobre ellos. Incluso a quien participa en desarrollos de software libre le cuesta en muchos casos explicar asuntos tan básicos como la forma de tomar decisiones, o los mecanismos de planificación y adaptación a las necesidades de los usuarios. Y en general, hay muy pocos datos cuantitativos sobre el software libre. Hasta hace un par de años ni siquiera había, por ejemplo, estimaciones aproximadas de la cantidad de código fuente libre desarrollado.

Y precisamente la ausencia de estos datos cuantitativos es lo más sorprendente, porque el software libre, por su propia naturaleza, permite realizar muchos tipos de estudios de forma completamente reproducible, dado que los datos necesarios están accesibles para cualquiera que los quiera mirar. Todo el código fuente (en muchos casos incluso su historia completa en un CVS), los informes de error (en los proyectos que mantienen un sistema de seguimiento de errores), los mensajes que intercambian los desarrolladores, y muchas cosas más están disponibles en Internet. Es probablemente la primera vez en la historia del software en que tenemos (literalmente) decenas de millones de líneas de código que usan millones de usuarios, y que están disponibles para que cualquiera las estudie... ¿No te parece excitante?

## Cinelerra

A finales de 2001 hubo una mala noticia en el mundo de la edición de vídeo con software libre: los productores de Broadcast2000, un editor no lineal de vídeo con capacidades profesionales, dejaban de mejorarlo. Afortunadamente, el verano de 2002 compensó con creces esta decepción: Broadcast2000 volvía, reconvertido en Cinelerra[1], con nuevas capacidades y con ganas de quedarse. De nuevo volvemos a tener un estupendo producto libre para editar películas en nuestras cajas GNU/Linux. Si no lo has probado, échale un vistazo. Es simplemente espectacular. Con poco tiempo que le dediques puedes empezar a hacer tus pinitos en la edición de vídeo. Y con los precios ridículos a que se han puesto las tarjetas digitalizadoras (y los precios razonables que empiezan a tener las cámaras DV y las tarjetas Firewire) cada vez es más viable tomar cualquier vídeo y retocarlo a gusto en el ordenador. Al menos la tentación de hacerlo una vez puede ser arrolladora...

[1] `http://heroinewarrior.com/cinelerra.php3`

## El entorno legal

Este año ha habido mucho movimiento en legislación. Por supuesto, está la famosísima LSSI, de la que no voy a hablar. Pero además, y más cercanas al software libre (en cuanto a efectos) tenemos la propuesta de directiva europea sobre patentes de programación, y la directiva sobre copyright (de las que nos ocupamos en el grupo ProInnova.

La propuesta de directiva que pretende introducir patentes de programación en Europa fue propuesta por la Comisión Europea en febrero. La parte positiva es que hay un movimiento a nivel europeo que está explicando los problemas que causaría a la innovación en la industria del software, y de forma muy particular al mundo del software libre. La parte negativa es que, hoy por hoy, esto no ha sido suficiente para parar el proceso de esta propuesta. Los primeros meses de 2003 serán decisivos, con su discusión en el Parlamento Europeo y con la toma de postura en el Consejo de Europa. Cualquier ayuda para seguir explicando lo que ganaremos si mantenemos Europa como una zona libre de patentes de programación es más que bienvenida.

La directiva europea sobre copyright, que ha pasado prácticamente desapercibida, presenta preocupantes problemas para el desarrollo de software libre en ciertos campos especialmente importantes (en general, todos los que tienen que ver con reproducción de material sujeto a derechos de autor), y para la conservación del derecho de copia privada (entre otros). Aún se puede ayudar a que la transposición de esta directiva a legislación nacional sea lo menos perjudicial posible, pero el tiempo corre y las presiones son muchas...

## OpenCourseWare

Sólo unas palabras para esta iniciativa del MIT[2]. A primera vista no consiste más que en poner, disponibles para el público en general en Internet, todos los materiales de las clases que se dan allí. Nada que no estén haciendo ya muchos profesores por todo el mundo. ¿Dónde está lo novedoso?

Echando un segundo vistazo, sin embargo, uno se da cuenta de que el MIT es el MIT. De que sus cursos no son cursos cualquiera, y sus profesores o sus alumnos tampoco. Y de que esta iniciativa no es una idea voluntarista de un profesor aislado, sino un esfuerzo coordinado y con todo el apoyo institucional, con financiación, con planificación, y con vocación de ofrecer de manera uniforme, relacionada y útil todo este material.

Mi opinión, después de estos dos vistazos: nunca los materiales docentes de la universidad volverán a ser lo mismo. Pero el tiempo dirá...

---

[2] `http://ocw.mit.edu/`

## Ahora, a ver qué tal el 2003...

Sin duda me he dejado muchos temas importantes (entre ellos, la gran actividad de miles de grupos de promoción del software libre por todo el mundo). Ésta tampoco pretende ser una lista exhaustiva, ni mucho menos. Ni siquiera es lo que más me ha impactado durante este año. Es solamente lo que más recordaba ahora, cuando estaba escribiendo. Seguro que tu lista es distinta, al menos igual de importante y probablemente mucho más valiosa. Pero aquí queda esta, por si te ha apetecido leerla...

¿Qué novedades nos traerá el 2003? Habrá que verlas, pero seguro que serán muchas, excitantes, y sorprendentes... ¿Quién se atreve a predecirlas?


Gracias a quien me sugirió este tema. Desgraciadamente, no recuerdo su nombre.

# Las piedras en el camino

Si hay alguien a quien los informáticos tememos, ése seguro que es un abogado. Por lo menos contamos con el (triste) consuelo de que a ellos les pasa lo mismo con nosotros. Bromas aparte, los mayores obstáculos que ha de superar el software libre en un futuro cercano no son problemas técnicos, ni de aceptación, ni tan siquiera económicos. El mayor problema tiene un claro caracter legal. Y así de perdidos estamos.

Para aclarar un poco los diferentes conceptos, la lectura de *Patentes, marcas o derechos de autor* es un buen comienzo. Trata, desde el punto de vista de un informático, los diferentes conceptos que atañen al software libre y, a grandes rasgos, cuáles son *benévolos* con él y cuáles ciertamente suponen un serio riesgo.

Sin duda, uno de los mayores riesgos es precisamente la posibilidad de que se permitan las patentes sobre programas. En las altas instancias a nivel europeo existe desde hace unos cuantos años un debate sobre el tema que se resume convenientemente en *Consulta de la Comisión Europea sobre patentes de software*. En él, el lector podrá encontrar información sobre por qué son perjudiciales para la industria del software (¡no sólo para el software libre!) y argumentan que los derechos de autor ya garantizan la innovación tecnológica en la dinámica industria del software como se puede demostrar haciendo un análisis histórico de este sector.

*Patentes de software, próximamente en esta pantalla* es un anexo al artículo anterior, actualizado a la situación de mediados de 2002 (el anterior databa de finales de 2000). Entonces las movilizaciones a nivel europeo en relación a este tema estaban ya a la orden del día y el desconocimiento sobre las implicaciones de la patentabilidad del software -aún siendo grande- era menor.

Y cambiando completamente de tercio, el último ensayo está dedicado a la LSSI, que propició un amplio debate -no exento de controversia- sobre las implicaciones que pudiera tener en el mundo Internet español. En *LSSI: ignorantes o censores* se muestran los puntos más conflictivos y por qué muchos de los aspectos que promulga la ley pueden suponer un retroceso en las libertades de los ciudadanos.

# Patentes, marcas o derechos de autor

Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 23, pág. 12-13, Noviembre de 2002

En los primeros tiempos de la informática, los usuarios se relacionaban con los fabricantes de ordenadores y de software fundamentalmente mediante contratos específicos. En los años 60, con el abaratamiento de los costes de los ordenadores y su consiguiente difusión se pasó a aplicar la legislación sobre los derechos de autor al software. Ahora se está intentando, fundamentalmente por la presión de las grandes multinacionales americanas, que la industria del software pase a regirse en base a los derechos de patentes. ¿Cuál es la diferencia entre patente y derecho de autor? ¿Qué tienen que ver las marcas comerciales con todo esto? Sin ser un experto en este tema, voy a intentar responder a algunas de estas preguntas.

## Patentes

¿Qué es una patente? En una definición, seguro que poco rigurosa, se trata de un monopolio temporal sobre una tecnología. Desde el punto de vista oficial se consienten y protegen estos monopolios porque supuestamente son el único mecanismo que garantiza que la inversión realizada en la investigación y desarrollo de un producto, técnica, etc. pueda recuperarse. El monopolio se implementa concediendo al titular de la patente el derecho a explotar ese producto, técnica, etc. en exclusiva. Eso sí, el derecho tiene algunas limitaciones: tiene una duración determinada (veinte años), debe ser explotada, etc.

¿Cómo se obtiene una patente? En general cada país dispone de una *oficina de patentes* encargada de otorgar ese derecho exclusivo en su territorio. En el caso de España es la Oficina Española de Patentes y Marcas (OEPM) que es un Organismo Autónomo del Ministerio de Ciencia y Tecnología. Algunos países además se han agrupado para reconocerse automáticamente esos derechos, ya que la legislación sobre propiedad industrial e intelectual es una de las armonizadas del mundo. Así, se ha creado por ejemplo la Oficina Europea de Patentes (EPO, *European Patent Office*). La EPO no depende de la Unión Europea. De hecho es un organismo formado también por países que no forman parte de la Unión (Suiza, Turquía, etc.) y que no depende de los presupuestos de la Unión, se autofinancia. Resalto este hecho, porque la fuente de ingresos principal de la EPO es el resultado de las tasas que cobra a los solicitantes de patentes, lo cual les motiva mucho a fomentar las patentes.

¿Quién defiende una patente? En contra de lo que mucha gente cree, las oficinas de patentes, como la EPO, no tienen la misión de defender las patentes. Se trata de un simple registro que certifica la fecha en la que se ha realizado

la solicitud de una patente y que realiza (y cobra) una serie de estudios para comprobar que ese producto no se haya patentado con anterioridad, o que no estuviese permitido patentarlo. Es decir, las oficinas de patentes hacen una serie de comprobaciones antes de conceder una patente, aunque obviamente se pueden equivocar. Una vez concedida una patente la *carga de la prueba* recae sobre la parte demandante.

¿Cómo se revoca una patente? De igual forma que las oficinas de patentes no defienden una patente, tampoco la revocan de oficio una vez concedida. Es decir, alguien que pretende demostrar que una patente no es legal tiene que probarlo ante las autoridades judiciales, a no ser, claro, que todo un país deje de reconocer las patentes, como por ejemplo hizo EEUU al independizarse del Reino Unido. Lo que hoy, curiosamente, no consiente a países como Sudáfrica, ni siquiera para salvar a su población del SIDA.

¿Cómo afectan las patentes a la industria del software? De una forma peligrosa. De hecho constituye, en opinión de gran parte de la comunidad informática, uno de los problemas más graves que afronta la industria en estos momentos. ¿Hubiera sido el desarrollo de la informática igual de rápida si se hubiese patentado el concepto de variable?

## Marcas

¿Qué es una marca? Resumidamente, no es más que una denominación, un signo o medio material de cualquier clase o forma que sirve para distinguir a un producto o servicio de otros similares. Las marcas tienen un dueño, pueden ser de hecho, uno de los activos principales de una compañía. Así, empresas como *Coca-Cola* o *El Corte Inglés* gastan gran parte de su presupuesto simplemente en cuidar y promocionar sus marcas. La titularidad de una marca tiene una duración indefinida (en contra de los mitos sobre su caducidad).

¿Cómo se obtiene una marca? Existen una serie de registros, dependientes de las administraciones públicas de cada país, que mantienen un registro de la marcas así como de los nombres comerciales, rótulos, etc. que en este artículo he agrupado bajo la sección marcas. En España es la misma oficina que en el caso de las patentes: la OEPM, puesto que se considera una marca como parte de la *propiedad industrial* de una empresa. De hecho, como decíamos antes, puede ser su propiedad más valiosa.

¿Cómo se reclama una marca? En el caso de que exista una disputa sobre una marca, es un juez el que tiene que dirimir a quién pertenece. Han existido muchos juicios de este tipo, siendo muy llamativo el caso de *Puma* en España. Sin embargo, si se piensa fríamente debería ser un problema menor, pues una empresa tiene una infinidad de posibilidades de denominar a su producto -basta tener la suficiente imaginación- y, desde luego, lo que promueven las marcas es un fin perseguible: una empresa ha creado una reputación y ninguna otra debería tener derecho a aprovecharla o a destruirla.

¿Cómo afectan las marcas a la industria del software? En principio yo creo que son neutrales y naturales. Siempre han existido marcas en la industria informática, tanto en el hardware como en el software. Microsoft es el nombre comercial de una empresa con marcas muy conocidas como Windows o Access, pero existen muchas otras empresas con infinidad de marcas, por ejemplo, IBM en los mismos segmentos ha competido con marcas como OS/2 o DB2.

## Los derechos de autor

¿Qué son los derechos de autor? Básicamente el reconocimiento público de que un determinado bien cultural (artístico, científico, etc.) ha sido producido por un determinado autor. La protección de los derechos de autor cae dentro de la legislación sobre propiedad intelectual, que es diferente de la propiedad industrial (patentes y marcas).

¿Cómo se obtienen los derechos de autoría? Para el caso de los bienes intelectuales: obras científicas, literarias, musicales, teatrales, cinematográficas, esculturas, pinturas, dibujos, grabados, litografías, *comics*, fotografías, etc. existen unos registros específicos. Así, en España, existe el registro de la Propiedad Intelectual que depende del Ministerio de Educación, Cultura y Deportes.

Es muy importante destacar, que a diferencia de las patentes y marcas, los derechos corresponden al autor por el mero hecho de la creación, a diferencia de la propiedad industrial, en la que se generan derechos exclusivamente mediante registro expreso y concreto de las patentes o de las marcas.

¿Cómo se revocan, reclaman y defienden los derechos de autor? De nuevo es responsabilidad del autor defender sus derechos. Muchos se han organizado en formas de sociedades de autores para defender colectivamente ese derecho. Es el caso en España de la Sociedad General de Autores y Editores (SGAE). Desgraciadamente, estas sociedades han acabado siendo meras gestoras de los *derechos de copia* (*copyright*) que se derivan del derecho de autoría, representando más a los editores que a los autores.

¿Cómo afectan los derechos de autor a la industria del software? Pues desde su aplicación a la industria del software en los años 60 han marcado la estructura de nuestra industria, de esa legislación y, por asimilación de otros mercados de propiedad intelectual, como el de la música, hemos acabado en un modelo de cobro por copia. La aparición en los últimos años de un modelo diferente, el software libre está haciendo tambalearse este mercado de licencias.

## ¿Qué deberíamos usar en la industria del software?

En mi modesta opinión, los autores tienen el derecho a que se reconozca que una determinada obra ha sido realizada por ellos. Que el derecho a ese reconocimiento implique tener derecho a limitar la forma en que se reproduce o modifica su obra es una afirmación que encuentro mucho más difícil de mantener.

Especialmente en el caso de productos intelectuales objetivos como el *software* (se puede certificar objetivamente que determinado algoritmo es más rápido, eficiente, etc.), pero también en el caso de los subjetivos como la música: el autor original siempre tiene la posibilidad de mantener y difundir *su versión*.

El fundamento principal de todos los problemas es que la mayoría de los bienes culturales no representan bienes materiales. Su reproducción en muchos casos tiene coste próximo a cero y su copia no tiene por qué suponer ningún menoscabo en el patrimonio de sus autores, su original sigue estando tal cual después de las copias. El problema estriba en que nuestra sociedad se ha acostumbrado a que la retribución a los productores de bienes intelectuales se realiza mediante la venta de copias, lo que por otra parte sólo garantiza unos ingresos razonables a una pequeña parte de los autores.

Por todo ello, creo que la protección de las marcas constituye el mecanismo idóneo para organizar el sector del software especialmente desde la perspectiva del software libre. IBM tiene por ejemplo una *marca* consolidada, se presuponen a sus productos unos estándares de calidad, bajas tasas de fallo, etc. Eso tiene claros beneficios para la empresa: se baraja en cualquier proyecto, es una de las primeras opciones de compra, etc. Las marcas además fomentan la competencia porque cualquiera puede crear *su marca* fácilmente.

Además, las marcas identifican a una empresa, no a un producto. Por ejemplo, Red Hat es la *marca* de una empresa americana que distribuye GNU/Linux. Bajo los principios del software libre cualquiera puede coger ese mismo software y revenderlo. De hecho, existen múltiples distribuciones basadas incluso en su mismo formato de paquete (el RPM). Los compradores son los que tienen la libertad de elegir qué marca quieren comprar, exactamente igual que deciden ir a realizar sus compras a *El Corte Inglés* o a *Carrefour*. Esta diferenciación en la marca defiende tanto a las empresas como a los productos. El caso de Red Hat Linux es claro; existen más empresas que distribuyen GNU/Linux. Igualmente, el hecho de que una determinada empresa empaquetara una versión incorrecta del compilador y el kernel de GNU/Linux de tal forma que no se podía compilar, no tiene por qué afectar al propio GNU/Linux, que funcionaba perfectamente en otras *marcas*.

Por el contrario, lo más peligroso son las patentes. De hecho, a mi modo de ver, las patentes, y en esto se diferencian de los derechos de autor como recalcaba antes, no premian la originalidad, la innovación. Trataré de ilustrarlo con un ejemplo: supongamos el caso de dos empresas que investigan por separado y en secreto en el mismo problema y llegan por separado a la misma solución. ¿Qué empresa tiene más derecho a tener la patente? Esto, que puede ser extraño en el caso de otras industrias, es muy corriente en el caso de la informática

La pregunta clave es: ¿Cuántas patentes estarás infringiendo sin saberlo en el siguiente programa que escribas?

## Más información

Como siempre, las opiniones expresadas en este artículo corresponden a las pocas luces del juntaletras que lo firma. Otras opiniones se pueden encontrar en muchos sitios. En particular, podrás encontrar información menos crítica con la idea de la patentabilidad del software en estas URLs:

- Oficina Española de patentes y marcas: `http://www.oepm.es`
- Oficina Europea de Patentes: `http://www.epo.org`
- Oficina Mundial de la Propiedad Intelectual: `http://www.wipo.int/`

Y argumentos mucho mejores que los míos en contra de la patentabilidad del software en estas otras:

- FreePatents: `http://www.freepatents.org/`
- ProInnova: `http://proinnova.hispalinux.es/`
- Eurorights: `http://uk.eurorights.org/`

# Consulta de la Comisión Europea sobre patentes de software

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 4, pág. 12-13, Diciembre de 2000

El tema de las patentes de software, que tanto puede afectar al desarrollo del software libre, está siendo tratado a nivel europeo. Por ahora, la Unión Europea es oficialmente un territorio libre de este tipo de patentes, pero esta situación puede cambiar con la nueva directiva que está preparando la Comisión Europea. Como parte de esta preparación ha tenido lugar un proceso de consultas (que terminó el pasado 15 de diciembre[1]) en el que se ha invitado a participar a los ciudadanos, las instituciones y las empresas europeas. Puede encontrarse más información sobre estas consultas en ([2]). También pueden verse las contribuciones canalizadas a través de EuroLinux en ([3]).

A continuación podéis leer mi contribución personal a este proceso de consultas, que detalla los motivos por los que creo que Europa no debería considerar los programas de ordenador como parte de las tecnologías susceptibles de ser patentadas.

## Introducción

Una patente da monopolio sobre una tecnología. Pero es bien sabido que los monopolios producen ineficiencias económicas y sus costes sociales no son normalmente despreciables. Por ello cualquier extensión de las áreas que cubre la legislación sobre patentes (como la extensión a los programas de ordenador sobre la que se está discutiendo) debería ser estudiada con cuidado. Cualquier beneficio debería ser probado y ponderado frente a estos costes e ineficiencias.

Las patentes se promueven normalmente como mecanismos para mejorar el desarrollo tecnológico en un área dada y para ayudar a los innovadores a que consigan suficiente capital para convertir sus ideas en productos. En el caso específico del software, la legislación sobre derechos de autor y la propia dinámica de la industria del software han sido suficientes para conseguir una historia notable de rápida innovación tecnológica y buena consecución de fondos. No hay evidencia de que las patentes sobre programas de ordenador mejoren esta historia. Por el contrario, hay evidencias de varios problemas que deberían solucionarse para

[1] N. del E.: la fecha referida es el 15 de diciembre de 2000

[2] `http://www.europa.eu.int/comm/internal_market/en/intprop/indprop/softpaten.htm`

[3] `http://petition.eurolinux.org/consultation`

poder mantenerla en el caso de que se introduzcan las patentes de software como un nuevo factor.

Como efecto colateral, las patentes de software son claramente un gran peligro para la industria del software libre. Cualquier área económica que esté libre de patentes de software tendrá ventajas competitivas para esta industria. En el caso de que el software libre florezca en los próximos años, este efecto será de gran importancia para la economía y para el desarrollo tecnológico.

## ¿Dónde están los beneficios que recibe la sociedad?

Cuando se va a estudiar el asunto de las patentes de software es muy importante recordar los fundamentos de esta discusión. La legislación sobre patentes no es un tipo de ley natural, sino un mecanismo usado por las sociedades para mejorar la velocidad de desarrollo tecnológico y para asegurar que no se ocultan las técnicas importantes de forma que puedan ser incorporadas al corpus de conocimiento público.

Sin embargo, estos beneficios no son gratuitos: tienen muchos costes para la sociedad, principalmente en forma de ineficiencias económicas y de inaccesibilidad a nuevos aparatos tecnológicos por gran parte de la población. Estos costes vienen del hecho de dar a un individuo o a una empresa el monopolio del uso comercial de la técnica patentada. El efecto de los monopolios en la economía ha sido bien estudiado por los economistas y su impacto, tanto en el desarrollo tecnológico como en el económico, puede ser realmente peligroso. Quien tiene una patente o un grupo de patentes relacionadas entre sí puede bloquear el desarrollo de ramas completas de una tecnología dada o imponer esquemas de licencia que fuercen a los fabricantes a vender sus productos a precios inadecuados para grandes sectores de la sociedad.

A pesar de estos efectos negativos, en ciertos sectores los efectos positivos de las patentes pueden ganarles en peso, siendo el efecto total resultante positivo para la sociedad. Por ello el problema que tenemos es decidir si en el caso específico del software este balance es positivo o negativo. La sociedad debería estar interesada en tener patentes de software sólo en el caso de que los beneficios sean claros y cuando se miden, sean de importancia suficiente como para producir un resultado positivo. Sólo en ese caso debería la sociedad considerar la posibilidad de prohibir a los individuos y a las empresas que usen libremente las técnicas que quieran, forzándoles a conseguir licencias de patentes.

No se encuentra en conocimiento del autor ningún estudio detallado y fiable que muestre esos beneficios en el caso del software. Tampoco ningún resultado de una investigación que muestre que el impacto neto de las patentes de software sobre la economía o sobre el desarrollo tecnológico es positivo. Por lo tanto, en ausencia de evidencias de tales beneficios y tal efecto positivo total, las patentes de software ni siquiera deberían ser consideradas. En el caso de que estos beneficios sean apreciables en el futuro, puede revisarse esta decisión.

## Relación entre patentes y desarrollo tecnológico en el caso del software

La industria del software es realmente dinámica. La barrera de entrada es muy baja en campos que tecnológicamente están en el frente de onda y es posible convertir ideas en productos con relativamente pocos recursos comparando con otras industrias. Por el contrario, en los campos donde la tecnología ya ha madurado hay fuerzas que normalmente causan la aparición de monopolios. Casi en cualquier nicho de software maduro hay un producto que tiene una fracción realmente grande del mercado.

Una barrera de entrada tan baja asegura que haya una fuerte competencia entre los innovadores. Ésa es la principal razón por la cual la velocidad de desarrollo en la industria del software es tan alta. Por otro lado, la legislación de derechos de autor asegura que los desarrollos que hace un innovador no pueden ser usados directamente por su competencia. El retraso con el que otras empresas pueden desarrollar sus propios productos es suficiente para asegurar la financiación al primer desarrollador, si es capaz de entregar un producto razonable. Conseguir dinero capital riesgo no es uno de los problemas fundamentales del desarrollo de software. Por el contrario, tenemos muchos ejemplos en los que encontrar recursos no ha sido un gran problema, como la cantidad de fondos conseguidos por la industria de Internet en la última década, que han sido dedicados principalmente a desarrollar software. La introducción de patentes de software incrementaría la cantidad de recursos que necesita un innovador para poder hacer nuevos productos. Necesitaría nuevos fondos para hacer estudios de patentes sobre su software, para pagar licencias en caso de que su software sea alcanzado por una o más patentes (incluso si no están relacionadas con las mejoras que introduce el producto) y para hacer provisiones frente a los previsibles gastos de litigar con dueños de patentes (incluso si la infracción de esas patentes no está clara).

La situación monopolística que se alcanza en muchos nichos de software cuando la tecnología madura es un problema conocido en esta industria y una barrera para la innovación en esos nichos. Los nichos de sistemas operativos, navegadores de web o aplicaciones ofimáticas son casos bien conocidos[4]. La introducción de patentes de software sólo podría reforzar esos monopolios. Además de su monopolio en el mercado, las empresas que tengan suficientes recursos podrían conseguir también un monopolio de la tecnología simplemente comprando patentes en su nicho. Cuando alcanzasen ese monopolio, podrían parar completamente la innovación en él realizada por terceras partes negándose a negociar licencias de sus patentes. Por supuesto, eso reforzaría su producto como la única opción.

Por estas y otras razones, el impacto de las patentes de software sobre el desarrollo de software y sobre la mejora de las tecnologías del software es cla-

[4] Nota del editor: El lector interesado en profundizar en estas cuestiones puede leer el artículo *Software libre, monopolios y otras yerbas* incluido en esta colección.

ramente negativo. No hay ningún estudio en conocimiento de este autor que muestre un impacto positivo de las patentes sobre el desarrollo tecnológico en el caso específico de las tecnologías de software.

## Impacto sobre el software libre

El impacto de las patentes sobre el software libre (o de código abierto) es, por su propia naturaleza, realmente negativo, e incluso peor que en el caso de otros tipos de software (como el software propietario). Hay tres características del software libre que explican este efecto negativo específico:

- Disponibilidad del código fuente. El código fuente siempre está disponible para su estudio y escrutinio en el caso del software libre. Eso significa que todas las tecnologías de software que se usan están completamente expuestas a un análisis de patentes. Si una empresa tiene que considerar la posibilidad de luchar en un juicio por infracción de patente, la exposición del código fuente no es la mejor estrategia posible. Las empresas querrán dificultar lo máximo posible las querellas por infracción de patente. Eso les forzaría a no publicar el código fuente de sus aplicaciones (ya sean tanto aplicaciones producidas como usadas por esas empresas).
- Imposibilidad de negociar licencias. El software libre puede copiarse y redistribuirse sin restricciones. Puede ser modificado e incorporado en otros productos libres. Por lo tanto, no hay ningún punto único de distribución como ocurre en el caso del software propietario. Eso hace que sea realmente difícil encontrar un esquema para negociar licencias para el uso de patentes en programas libres y es muy poco probable que se concedan licencias de muchas patentes para su uso en programas libres.
- Impacto en pequeños desarrolladores. El software libre se desarrolla en muchos casos por empresas muy pequeñas y desarrolladores individuales, con mucha frecuencia en su tiempo libre. Se reciben contribuciones de mucha gente de todo el mundo. El trabajo de examinar todo el código producido y todas las contribuciones recibidas buscando posibles usos de tecnologías patentadas está completamente fuera de las posibilidades de esos desarrolladores. Por lo tanto, si hay que realizar estudios de infracción de patentes antes de distribuir software (debido al riesgo de ser acusado de infracción de patente) muchos de estos desarrolladores no podrán producir productos con software libre. Incluso si no usan ninguna tecnología patentada.

Por lo tanto, la promoción del software libre es absolutamente incompatible con la introducción de las patentes de software y no es casualidad que la comunidad del software libre sea una de las más activas en la lucha contra las patentes de software. Por otro lado, cualquier área económica que pueda mantenerse libre de patentes de software será un buen lugar para establecer negocios basados en software libre. Esta ventaja competitiva no será despreciable si la industria del software libre alcanza el potencial que muchos expertos esperan.

## Conclusiones

Las patentes de software, como cualquier tipo de patente, tienen efectos negativos para la sociedad en su conjunto. En otras industrias pueden producir suficientes beneficios que los contrapesen, pero éste no es el caso de las patentes de software. No mejoran la velocidad de desarrollo de software y pueden dañar a los pequeños (pero muy productivos) innovadores. En el caso del software libre el impacto de las patentes de software es especialmente dañino.

Por todas estas razones, no veo ninguna necesidad de introducir patentes de software, ni ningún beneficio global para la industria del software o para la sociedad en su conjunto. Por el contrario, los efectos de su introducción serían negativos en muchos aspectos.

# Patentes de software, próximamente en esta pantalla

Jesús M. González Barahona

Publicado originalmente en la revista TodoLinux
Número 20, pág. 12-13, Mayo de 2002

En febrero[1] la Comisión Europea hizo pública su propuesta de Directiva sobre Patentabilidad del Software. Si esta propuesta acaba siendo aprobada, las patentes de software serán válidas en Europa y tendrán unas características muy similares a las que ya tienen en EEUU. Éste sería el cambio legislativo más importante para la industria del software desde los años 60, cuando se decidió aplicar a los programas de ordenador la legislación de derechos de autor. Su impacto en el sector sería, sin duda, enorme, y miles de millones de euros cambiarían de bolsillo. Se acabaría la situación actual donde quien desarrolla un programa está seguro de que puede comercializarlo y organizar un negocio a su alrededor, donde quien tiene una buena idea es libre de realizar un programa que la haga realidad. Por el contrario, los que posean los derechos sobre patentes de software tendrán el control sobre campos clave de la economía y de la innovación, y recibirán ingentes transferencias de fondos de los productores de programas.

Este cambio tan crucial para el futuro de un sector que mueve una parte considerable de la economía europea se va a realizar sin que la mayor parte de los actores implicados se hayan enterado de él ni de sus consecuencias para su futuro. Y en contra de la opinión de la inmensa mayoría de los que sí se han enterado de estas implicaciones.

Aún es posible tratar de evitar esta situación. Pero el tiempo se agota...

## La propuesta de Directiva sobre la Patentabilidad del Software

El 20 de febrero de 2002 la Dirección General para el Mercado Interno de la Comisión Europea (lo más parecido que tenemos a un Gobierno Europeo) hizo pública su propuesta de Directiva sobre la Patentabilidad del Software. Esta propuesta supone la admisión de las patentes sobre programas, cambiando radicalmente la situación actual (la Convención Europea de Patentes especifica claramente que los programas de ordenador no están en el ámbito de lo patentable).

En la actualidad, la Directiva está siendo negociada según el proceso de codecisión, entre el Parlamento Europeo y la Comisión Europea. Para entrar en vigor, ha de ser aprobada por el Consejo Europeo (compuesto por representantes

[1] Nota del editor: febrero de 2002

de los estados miembros de la Unión Europea) y por el Parlamento Europeo. En caso de ser aprobada, se *transpone* en legislación nacional (y es, por tanto, de obligado cumplimiento) unos meses después, previa votación en los parlamentos nacionales (aunque este último paso es poco más que un mero trámite).

En el momento de escribir esta nota, sólo el Gobierno de Francia ha mostrado su oposición a la Directiva. Por ahora, el resto de los gobiernos nacionales no se han pronunciado, y la comisión del Parlamento Europeo encargada de la negociación con la Comisión Europea aún no ha hecho públicos sus comentarios.

La propuesta de Directiva ignora los informes oficiales realizados en países como Francia y Alemania, que muestran el impacto negativo de las patentes de software sobre la innovación. También ignora los resultados públicos de la consulta realizada por la propia Comisión Europea a finales de 2000, en la que una inmensa mayoría de las contribuciones desaconsejaban la introducción de las patentes de software. Y, por supuesto, también ignora las más de 120.000 firmas que ha recogido EuroLinux contra las patentes de software.

Pero lo que es más grave, esta Directiva coge completamente por sorpresa a la mayor parte de los implicados. Ni los productores ni los usuarios europeos de informática tienen, en general, idea de lo que está ocurriendo, ni del enorme cambio que esta Directiva, caso de aprobarse, va a suponer para sus negocios. La mayoría de ellos no ha tenido la más mínima oportunidad de pronunciarse informadamente, y si el trámite de la propuesta de Directiva no la modifica radicalmente, se van a encontrar con el mayor cambio de reglas que ha sufrido la industria europea del software desde que existe, sin prácticamente comérselo ni bebérselo.

## Implicaciones de la propuesta de Directiva

Si la propuesta de Directiva termina convirtiéndose en legislación, Europa perderá una oportunidad de oro de situarse como el territorio desarrollado con menos obstáculos a la innovación en software. Aunque la Comisión Europea ha maquillado el lenguaje de la Directiva de modo que parezca que solamente algunos programas serían patentables, una cuidadosa lectura de la parte normativa de la propuesta muestra claramente cómo será posible patentar cualquier programa de ordenador incluyendo, posiblemente, modelos de negocio y cualquier actividad que pueda realizarse con la ayuda de un ordenador.

Las patentes de software son (como cualquier otra patente) básicamente un monopolio sobre una tecnología. La duración de ese monopolio es de 20 años, durante los cuales el dueño de la patente decide quién puede usar esa tecnología en sus programas. Para permitir ese uso, puede pedir la compensación económica que quiera (usualmente un porcentaje de los ingresos por ventas). O, si lo prefiere, puede impedir completamente su uso, por ejemplo para fortalecer el monopolio en un nicho de mercado.

La justificación de las patentes está en su efecto beneficioso sobre la innovación. Sin embargo, en informática, la propia situación actual del mercado, que hace imprescindible la innovación para poder competir, ya proporciona suficientes alicientes para la innovación (y desde luego, entre todas las industrias, difícilmente podría decirse que es la informática la que más ayudas a la innovación precisa). Por ello, es cuando menos dudoso que las patentes de software ayuden en algo a la industria. Por el contrario, conllevan muchos inconvenientes y cambian las reglas en un sentido que en general no beneficia al interés del sector.

Además, el hecho de que los programas de ordenador sean fundamentalmente una forma de información hace que sea imposible diferenciar entre la difusión del conocimiento (que es un objetivo que persigue el sistema de patentes) y su uso comercial. Cuando un programador implementa una técnica patentada, está, a la vez, codificando ese conocimiento (lo que está permitido e incluso promovido por el sistema de patentes) y construyendo una herramienta susceptible de usarlo comercialmente (lo que está terminantemente prohibido sin el permiso del dueño de la patente). Esta dualidad se resuelve, cuando se aceptan las patentes de software, prohibiendo de facto ambas actividades. Pero al prohibir la expresión del conocimiento informático en forma de programas, lo que se hace es poner trabas al avance en las tecnologías relacionadas con la informática, consiguiendo precisamente el objetivo contrario al que teóricamente se persigue.

Desde un punto de vista más concreto, el desarrollo de la sociedad de la información en Europa se vería muy perjudicado en caso de que la actual propuesta de Directiva prospere. Las empresas europeas perderían competitividad frente a EEUU (al perder la ventaja competitiva que les da hoy día el no tener que preocuparse de las patentes), y deberían rediseñar su modelo de negocio de forma que tengan en cuenta el pago de patentes y la provisión de fondos para posibles querellas por infringir oscuras patentes (como una de las más de 30.000 ya concedidas, con dudoso respeto a la legalidad vigente, por la Oficina Europea de Patentes).

En el campo concreto del software libre, es importante comprender que las patentes de software no son más que un obstáculo a su desarrollo. El software libre puede desarrollarse bien en un entorno donde los derechos de autor son los que marcan la *propiedad* sobre los programas. Basta con realizar desarrollos independientes para que nadie pueda reclamar derechos sobre ellos. Pero si las reglas cambian, y las patentes pueden aplicarse a los programas, cualquier proyecto de software libre puede verse amenazado por el dueño de una patente. Además, incluso si ese dueño tiene voluntad de negociar licencias de explotación, difícilmente podrá llegar a acuerdos con proyectos de software libre, que no perciben ingresos directamente de los usuarios, y por lo tanto no pueden pagar un porcentaje sobre ventas. Por ello, es absolutamente imposible promover el software libre, e incluso pretender neutralidad con respecto a él, y a la vez promover las patentes de software.

## ¿Qué podemos hacer?

Ante esta situación, ¿qué podemos hacer, si queremos mantener a Europa como un territorio libre de patentes de software? Por separado, probablemente no mucho, pero una acción coordinada aún podría frenar el trámite de esta propuesta de Directiva. En España, puedes consultar las páginas de ProInnoiva [2], y unirte a sus acciones. Además, puedes tratar de informar a quien tomará las decisiones, por ejemplo mediante:

- Comunicaciones a los representantes que tenemos en el Parlamento Europeo, en las Cortes nacionales o en los Parlamentos autonómicos, para que discutan este problema a nivel político, y defiendan los intereses de los que les votamos.
- Comunicaciones a las diferentes administraciones (nacionales, autonómicas, locales), informándoles del problema, y aconsejándoles que tomen las medidas que estén en su mano para ayudar a frenar el proceso de esta propuesta de Directiva.
- Resoluciones de cualquier tipo de organización relacionada con las tecnologías de la información, urgiendo a mantener Europa como un territorio libre de patentes. Entre estas organizaciones están las empresas, los grupos de usuarios, las asociaciones profesionales, etc., que estén a nuestro alcance.
- Información, información, información. Cuando se explican las consecuencias de estar sometidos a las patentes de software, en general se ven rápidamente sus inconvenientes. Procura, en la medida de lo posible, que todo el mundo en tu ámbito esté informado sobre sus peligros, y sobre lo que nos jugamos a corto plazo.

En los próximos meses nos vamos a jugar el futuro de la innovación en software en Europa y sobre todo de la libertad de innovación. Si no conseguimos frenar a los que pretenden cambiarnos las reglas sin siquiera tenernos en cuenta, nunca más podremos escribir, comercializar o usar un programa sin miedo a tener problemas legales.

¿Es ése el futuro que quieres?



[2] http://proinnova.hispalinux.es

# LSSI: Ignorantes o censores

Javier Candeira y Vicente Matellán Olivera

Publicado originalmente en la revista TodoLinux
Número 13, pág. 12-13, Noviembre de 2002

En el momento de escribir este artículo el debate de la LSSI ha comenzado en el Parlamento, han convocado a algunos expertos para escuchar sus opiniones (de Kriptópolis y de la Asociación de Internautas) y pronto sabremos qué deciden sus señorías. Decidan convertir la propuesta en ley o no, creemos que conviene aclarar ciertos aspectos.

## Introducción

Una de las excusas de muchos de los gestores de las empresas *punto com* sobre el fracaso de sus negocios y las pérdidas de las inversiones realizadas en ellos es que el comercio electrónico no ha crecido al ritmo esperado. El motivo de esa lentitud es, según ellos, que la red es *insegura*. No lo compartimos, no tenemos ningún problema en comprar a través de la red, o al menos no más preocupación que al pagar en un restaurante, sin embargo compramos poco por la misma razón por la que compramos poco por catálogo: porque preferimos ir a una tienda, o a un gran almacén. Dicho de otra manera, no compramos más porque no queremos.

Sin embargo, los comerciantes han conseguido convencer al ministerio de turno, esta vez al de Ciencia y Tecnología, de que sus males se deben a la falta de regulación. El ministerio diligentemente ha propuesto (de momento está en fase de anteproyecto) una ley denominada: *Ley de Servicios de la Sociedad de Información y Comercio Electrónico*, más conocida por las siglas LSSI.

Otro posible motivo a nuestro entender más real que la seguridad, es la ilusión reguladora de algunos sobre la red. Nos explicamos: ¿Por qué tenemos sólo 6-7 cadenas de televisión? ¿Por qué tenemos 10-30 cadenas de radio en una ciudad de tamaño medio? ¿Acaso el espectro radioeléctrico no permite tener más? Por supuesto que lo permite, los que no lo permiten son los afortunados poseedores de una licencia, que con un enorme poder mediático, impiden que se concedan más y además se quejan por tener que pagar un canon. Su argumento principal para que no se concedan más licencias es *que no serían viables económicamente*, lo que no deja de tener su gracia en un modelo capitalista como el nuestro: es la administración la que decide quién sobrevive en el mercado en vez de ser el propio mercado...

Nos tememos que algunos quieran exportar ese modelo a Internet, nos tememos que algunos quieran colocarnos *licencias* para poder poner un servidor en la red, y lo que es peor, una vez puestas ya estamos sólo a un paso de poder limitar su número con cualquier criterio. La LSSI es un primer paso en ese camino. Los registros de proveedores de información son un enorme paso en esa dirección,

el no aceptar cualquier petición de registro automáticamente es inmediato una vez establecido el registro: rápidamente alguien argumentará que no puede ser automático para evitar a los de siempre, los terroristas y los pederastas, como si los hubiéramos inventado en la Red.

## La alarma

Por todo esto y por algunas cosas más, el intento regulador de Internet que propone la LSSI es preocupante. La alarma más intensa ante la propuesta de la LSSI la disparó Kriptópolis, revista independiente sobre criptología, seguridad y privacidad en Internet, que mantiene una campaña contra esa propuesta[1].

¿Cuál es el problema? Pues el más grave es que esta ley no pretende su aplicación únicamente al comercio electrónico, como se ha querido hacer ver. El objeto de estas normas es regular cualquier tipo de actividad que se realice a través de Internet. El concepto material en torno al que gira su ámbito de aplicación se denomina "servicios de la sociedad de la información" y, como veremos seguidamente, ahí no cabe sólo el, ya fallido, comercio electrónico. De esta forma, todo el que preste un "servicio de la sociedad de la información" se convierte automáticamente en prestador de servicios de la sociedad de la información y, consecuentemente, cae dentro del ámbito de aplicación de la Ley.

No queremos creer que los autores de esta ley (afortunadamente todavía en fase de borrador) la hayan hecho así por malvados, sino por desconocimiento. En cualquier caso, coincidimos con los analistas de Kriptópolis en que es un desaguisado que, de aprobarse en el Parlamento, habría borrado del mapa el derecho a la libertad de expresión en la red.

## La libertad de prensa

William Randolph Hearst, el magnate de los periódicos satirizado por Orson Welles en su Ciudadano Kane, solía decir que "la libertad de prensa es para los que tienen una" (prensa de imprimir, se entiende). Hoy cualquiera puede tener la suya. Cualquiera puede tener su página web, o escribir su opinión en la de otros.

Ahora que Internet nos permite a todos tener la libertad de expresarnos por muy poco dinero, o incluso de forma gratuita, sería una pena que nos fuera arrebatada por una ley diseñada para garantizar la legitimidad de los mercaderes que ofician detrás de un mostrador virtual o por los que quieren su parcela en el oligopolio de las licencias para transmitir información.

Y eso, eliminar la libertad de expresión, es exactamente lo que hace la LSSI, en el borrador tan duramente criticado por Kriptópolis. ¿Cómo? Pues mediante un proceso de tres pasos:

[1] `http://www.kriptopolis.com/lssi/`

- Haciendo que el "suministro de información por línea" sea objeto de la ley, que en principio sólo venía a armonizar el comercio electrónico en la UE. Tanto BarraPunto como Kriptópolis, El País como El Mundo, la web corporativa de Seat como cualquier página de Geocities, todos somos suministradores de información por línea y, por lo tanto, todos estamos sujetos de la misma forma a la LSSI, aunque no hagamos comercio electrónico.
- Convirtiéndonos a todos en policías. "todos los prestadores de los servicios de la sociedad de la información deberán cumplir las siguientes obligaciones: (.../...) supervisar el contenido de los datos e informaciones que constituyen el objeto del servicio de la sociedad de la información que prestan y realizar el control respecto de los hechos o circunstancias contenidas en aquéllos que pudiesen constituir actividades ilícitas".
  Esto, resumido, quiere decir que en BarraPunto nos haríamos responsables de supervisar los comentarios. Que cada proveedor de acceso al dar espacio en el servidor para alojar la página web de cada uno, deberá supervisar que todo lo que se pone en esas páginas es *lícito*. Este párrafo del borrador de ley se contradice con el artículo 15 de la directiva comunitaria: "no impondrán a los prestadores de servicios una obligación general de supervisar los datos que se transmitan o almacenen, ni una obligación general de realizar búsquedas activas de hechos o circunstancias que indiquen actividades ilícitas." Da igual. La directiva no es una ley, y la ley sí. Por eso es más preocupante lo que diga la ley.
- Por último, estableciendo que basta una autoridad administrativa para obligar a "suspender la transmisión, el alojamiento de datos, el acceso a las redes de telecomunicaciones o la prestación de cualquier otro servicio de la sociedad de la información". Esto quiere decir que la *autoridad administrativa* puede cerrar cualquier sitio web a su criterio. En el mundo del papel, para secuestrar un periódico hace falta la firma de un juez. Según esta ley, en el ciberespacio basta con una resolución administrativa.

Pero esto no es lo peor. A las resoluciones administrativas se une el miedo, o la cautela, de los proveedores de servicios. ¿Que no sé si un contenido puede ser ilegal o no? Pues lo quito, por si acaso. La ley exonera de responsabilidad a proveedores de alojamiento o almacenamiento, siempre que "no tengan conocimiento efectivo de que la actividad o la información almacenada es ilícita" o que "si lo tienen, actúen con diligencia para retirar los datos o hacer imposible el acceso a ellos".

## ¿Cómo nos afecta?

Imaginemos ahora el caso de un particular que tiene una página en un servidor, y recibe una denuncia. A la vez que este editor (porque es un editor, ya que ha publicado una página en Internet) recibe la denuncia, los responsables de su

proveedor de alojamiento reciben una copia, con la exhortación a quitar esa página de su servidor, pues es ilícita. ¿Qué creéis que harán los del centro de datos? ¿Qué creéis que les dirán sus jefes que hagan? La pregunta no es "¿qué deberían hacer si el mundo fuese justo?", pues ya sabemos que la respuesta es "esperar a la resolución judicial". La pregunta es ¿qué harán en el mundo real? Sí, el crudo mundo real, en el que *pedófilo*, *drogadicto*, *terrorista* o, simplemente, *delincuente*, son palabras tan fuertes con las que no quiere verse asociado ningún departamento de relaciones públicas de una empresa. La respuesta es obvia, la página o páginas completas de ese usuario se cerrarían inmediatamente.

Para que no tengamos libertad de expresión no hace falta que nos quiten la palabra a todos: basta con que se la quiten a uno. Y para asegurar la censura, no hace falta con ejercerla. Basta con poner en marcha su mecanismo. Y, si nadie se queja, el mal ya está hecho.

Por esta razón creo que todos los españoles deberíamos agradecer que se haya dado esta alarma sobre el presente borrador de la LSSI. A los que los que se quejan del alarmismo sólo se les puede contestar que la situación, tal y como la pintaba el borrador, era alarmante. Y lo que menos me preocupa es si esta situación se debiera a malicia, desconocimiento o incompetencia. El comportamiento responsable consiste en arreglarla. Lo irresponsable es negar que la situación sea alarmante.

Moncho Alpuente solía decir "la situación es alarmante, pero no preocupante, porque preocupándonos no vamos a llegar a nada". Al dar la voz de alerta y promover el discurso público, el especial de Kriptópolis ha prestado un gran servicio a la sociedad española y al futuro desarrollo de Internet. Falta ver a dónde nos lleva la posterior revisión de la ley y en qué acaba todo esto. El primer paso, ahora, está bien dado.

## ¿Qué podemos hacer?

Si viviéramos en EEUU la respuesta sería escribir a nuestro senador o congresista. Desgraciadamente eso no tiene mucho éxito en nuestro país, aunque Kriptópolis tiene un modelo en su web dirigido en este caso al Ministerio de Ciencia y Tecnología. La solución tradicional de rezar tampoco parece muy prometedora, aunque los internautas creyentes deberían ponerse a ello. Para el resto queda la opción del pataleo o la del asociacionismo. Nosotros nos inclinamos por este último para dar fuerza a los argumentos, en resumen: ¡No mires, únete!

## Para mentes inquietas

Si quieres leer más sobre este asunto:
`http://barrapunto.com/article.pl?sid=01/05/13/0233209`


`javier@canderia.com, vmo@barrapunto.com`

# Artículos Seminales

El fenómeno del software libre, como toda corriente de pensamiento tiene una serie de documentos y de autores que se consideran fundamentales. Algunos de esos documentos son los que recogemos en esta sección. Como el lector podrá comprobar, éstos son solamente una brevísima selección de los que podríamos denominar como artículos seminales del software libre. Hay ciertamente muchos más que, por el carácter limitado de esta obra (en tiempo y espacio), no hemos podido incluir.

En primer lugar nos encontraremos con el artículo titulado *Por qué el Software no debería tener propietarios* escrito por Richard Stallman. Este ensayo se considera la declaración de principios del movimiento del software libre y en el se expresan con claridad los motivos éticos por los que el *copyright* no debería aplicarse al software.

El segundo artículo *El derecho a leer*, también escrito por Richard Stallman, es una parábola de ciencia ficción que explica los peligros de ciertos derechos sobre la producción científica o literaria. Lo llamativo es que según ha ido pasando el tiempo (el artículo se publicó por primera vez en 1997) esa ciencia ficción cada día parece más real.

Por último, se ha incluido un artículo que plasma las ideas lanzadas en *Por qué el Software no debería tener propietarios*. Se trata de la *Definición de software libre* que hace la Free Software Foundation, una fundación creada por el propio Stallman para velar por la difusión de los valores y los programas del software libre. Existen otras definiciones sobre lo que es o no software libre (como por ejemplo, la la definición de software libre de Debian[2] o la de Open Source de la Open Source Iniciative[3]) pero ésta es sin duda la gran precursora de todas ellas y, en definitiva, en la que se basan todas las demás.

[2] http://www.debian.org/social_contract.es.html\#guidelines

[3] http://www.opensource.org

# Por qué el Software no debería tener propietarios

Richard Stallman

Traducido por Pedro de las Heras Quirós
Publicado en Español en la revista Novática
Número de 2001

La tecnología de la información digital contribuye a la sociedad haciendo que sea más fácil copiar y modificar la información. Las computadoras prometen hacer que esto sea más fácil para todos nosotros.

Pero no todo el mundo quiere que sea más fácil. El sistema de *copyright* asigna *propietarios* a los programas software, y muchos de estos *propietarios* pretenden negar el beneficio potencial del software al resto del público. Les gustaría ser los únicos que pueden copiar y modificar el software que utilizamos.

El sistema de *copyright* se desarrolló con la imprenta —una tecnología para producir copias en masa. El *copyright* funcionaba bien con esta tecnología porque sólo restringía a los productores masivos de copias. No coartaba la libertad de los lectores de libros. Un lector corriente, que no poseyese una imprenta, sólo podía copiar libros a mano, con pluma y tinta y pocos lectores fueron demandados por ello.

La tecnología digital es más flexible que la imprenta: cuando la información tiene un formato digital, ésta se puede copiar fácilmente para compartirla con los demás. Es esta flexibilidad la que no encaja con un sistema como el del *copyright*. Ésta es la causa de las cada vez más repugnantes y draconianas medidas que se están utilizando para obligar a cumplir el *copyright* del software. Ténganse en cuenta estas cuatro prácticas de la Asociación de editores de software (SPA[1]):

- Propaganda masiva diciendo que no se debe desobedecer a los propietarios para ayudar a los amigos.
- Ofertas a soplones para que informen acerca de sus compañeros y colegas.
- Redadas en oficinas y escuelas (con la ayuda de la policía), en las que se conmina a la gente a que pruebe que son inocentes, que no han realizado copias ilegales.
- Persecución (por parte del gobierno de los EEUU, a petición de la SPA) de gente como David LaMacchia, del MIT (Massachusetts Institute of Technology), por el simple hecho de no vigilar los servicios de realización de copias, y no censurar su uso.

Estas cuatro prácticas se asemejan a las utilizadas en la antigua Unión Soviética, donde toda máquina copiadora estaba vigilada por un guarda para evitar

[1] Software Publishers Association

copias prohibidas, y donde los individuos tenían que copiar la información secretamente y pasarla de mano en mano como *samizdat*[2]. Existe por supuesto una diferencia: el motivo por el que se controlaba la información en la Unión Soviética era político; en los EEUU el motivo es el lucro. Pero son las acciones las que nos afectan, no los motivos. Cualquier intento de bloquear la compartición de la información, sin importar el porqué, conduce a los mismos métodos y a la misma severidad.

Los propietarios esgrimen diversos argumentos para otorgarse el poder de controlar cómo usamos la información:

**Insultos** Los propietarios utilizan calumnias como *piratería* y *robo*, así como terminología pericial como *propiedad intelectual* y *perjuicio*, para sugerir una línea de pensamiento al público—una analogía simplista entre programas y objetos físicos.
Nuestras ideas e intuiciones acerca de la propiedad de los objetos materiales se centran en si es correcto o no quitarle un objeto a alguien. No se aplican directamente a la copia de los objetos. Pero los propietarios quieren que lo apliquemos también a la copia.

**Exageración** Los propietarios dicen que sufren *daños* o *pérdidas económicas* cuando son los usuarios los que copian los programas. Pero la copia no causa un efecto directo en el propietario, y no daña a nadie. El propietario sólo puede perder si la persona que hizo la copia hubiese pagado por una proporcionada por el propietario.
Reflexionando un poco, la mayor parte de esas personas no habrían comprado copias. Sin embargo, los propietarios calculan sus *pérdidas* como si todos y cada uno de los usuarios hubiesen comprado una copia. Esto es una exageración (por decirlo suavemente).

**La ley** Los propietarios muestran a menudo la situación legal actual, y las severas penas con las que nos pueden amenazar. En este enfoque está implícita la sugerencia de que la ley refleja una moralidad incuestionable, aunque al mismo tiempo somos instados a considerar estas penas como algo natural que no puede ser imputado a nadie.
Esta forma de persuasión no está diseñada para ser rebatida por un pensamiento crítico, sino que tiene la intención de reforzar una manera habitual de pensar.
Es elemental que las leyes no deciden entre lo que está bien y lo que está mal. Todo estadounidense debería saber que hace cuarenta años era ilegal en muchos estados que una persona de raza negra se sentase en la parte delantera de un autobús, aunque sólo los racistas dirían que sentarse allí era algo malo.

**Derechos innatos** Los autores reivindican a menudo una relación especial con los programas que escriben y afirman que, por lo tanto, sus deseos e intereses con respecto a los programas son mayores que los de cualquier otro. Mayores

[2] Nota del traductor: *Samizdat* es un vocablo ruso que significa *publicado por uno mismo*

incluso que los de todos los demás. (Normalmente, las compañías, y no los autores, son quienes poseen los derechos de autoría del software, pero se espera que pasemos por alto esta discrepancia).

A aquéllos que proponen esto como un axioma ético (*el autor es más importante que tú*) sólo puedo decirles que yo, un notable autor de software, digo que eso es una bobada.

Pero hay únicamente dos razones por las que el público en general podría sentir alguna simpatía con la reivindicación de derechos innatos:

Una razón es la analogía, cogida por los pelos, con los objetos materiales. Cuando cocino espagueti, pongo objeciones a que alguien se los coma, porque entonces no me los puedo comer yo. Su acción me perjudica tanto como a él le beneficia. Sólo uno de los dos puede comerse los espagueti, por lo que la pregunta es ¿quién?. La más mínima diferencia entre nosotros es suficiente para inclinar la balanza ética en uno u otro sentido.

Pero el hecho de que utilices o cambies un programa que yo escribí te afecta directamente a ti, y a mí sólo me afecta indirectamente. Si le das una copia a tu amigo, os afecta a ti y a tu amigo mucho más de lo que me afecta a mí. Yo no debería tener el poder de decirte que no hagas esas cosas. Nadie debería tenerlo.

La segunda razón es que se le ha dicho a la gente que los derechos innatos de los autores son una tradición incuestionable en nuestra sociedad.

Históricamente, es justamente al contrario. La idea de los derechos innatos de los autores se propuso y se rechazó contundentemente cuando se redactó la constitución de los EEUU. Por ello la constitución permite un sistema de *copyright*, pero no requiere uno. Por eso dice que el *copyright* debe ser temporal. También dice que el propósito del *copyright* no es recompensar a los autores, sino promover el progreso. El *copyright* recompensa en alguna medida a los autores, y mucho más a los editores, pero como una medida para modificar su comportamiento.

La auténtica tradición establecida en nuestra sociedad es que el *copyright* coarta los derechos innatos del público, y que esto sólo se puede justificar si es en beneficio de la sociedad.

**Aspecto económico** La última razón argüida a favor de los propietarios de software es que así se consigue producir más software.

A diferencia de los otros argumentos, éste al menos adopta un enfoque legítimo sobre el asunto. Se basa en una meta defendible, para satisfacer a los usuarios de software. Y es demostrable empíricamente que si se remunera adecuadamente la producción de un bien, se producirá más cantidad del mismo.

Pero el razonamiento económico tiene un defecto: se basa en la suposición de que la diferencia solamente radica en cuánto dinero tenemos que pagar. Asume que lo que se desea es la *producción de software*, sin importar si éste tiene o no propietarios.

La gente acepta fácilmente esta suposición porque concuerda con nuestra experiencia con los objetos materiales. Consideremos por ejemplo un bocadillo. Supongamos que el mismo bocadillo puede ser obtenido, bien de manera gratuita, o bien pagando. En este caso, la única diferencia entre ambos bocadillos es la cantidad que se paga por cada uno de ellos. El bocadillo tendrá el mismo sabor y el mismo valor nutritivo, sea comprado o no, y en ambos casos el bocadillo sólo podrá ser ingerido una vez. El hecho de que sea el propietario quien nos proporciona el bocadillo, no afecta directamente más que a la cantidad de dinero que acabaremos teniendo al final.

Esto es cierto para cualquier objeto material—el hecho de que tenga o no un propietario no afecta directamente a lo que es, o a lo que se puede hacer con él si se adquiere.

Pero el que un programa tenga o no propietarios afecta a lo que es, y a lo que se puede hacer con una copia si se compra. La diferencia no es sólo una cuestión de dinero. El sistema de propietarios de software alienta a éstos a producir algo, pero no a producir aquéllo que necesita realmente la sociedad. Y esto provoca una polución ética intangible que nos afecta a todos.

¿Qué necesita la sociedad? Necesita información que esté realmente disponible para sus ciudadanos. Por ejemplo, programas que la gente pueda leer, corregir, adaptar y mejorar, no sólo utilizar. Pero lo que normalmente distribuyen los propietarios es una caja negra que no podemos estudiar o modificar.

La sociedad también necesita libertad. Cuando un programa tiene propietarios, los usuarios pierden la libertad de controlar parte de sus propias vidas.

Y por encima de todo, la sociedad necesita alentar el espíritu de cooperación voluntaria entre sus ciudadanos. Cuando los propietarios de software nos dicen que la ayuda a nuestros vecinos es una forma de *piratería*, están corrompiendo el espíritu cívico de nuestra sociedad.

Por ello decimos que el software libre se refiere a las libertades, y no a la gratuidad[3].

El argumento económico que esgrimen los propietarios es erróneo, pero el problema económico general es real. Hay gente que escribe software de utilidad por el placer de escribirlo o por admiración y amor. Pero si queremos tener más software que el que esta gente escribe, necesitamos conseguir fondos para ello.

Hace ya diez años que los desarrolladores de software libre vienen utilizando varios métodos para buscar financiación, habiendo conseguido algunos éxitos. No es necesario hacer rico a nadie; el ingreso anual medio de una familia estadounidense, alrededor de los 35.000$, parece ser suficiente incentivo para muchos trabajos que son menos satisfactorios que la programación.

Durante varios años yo viví de realizar mejoras a medida del software libre que había escrito, hasta que una beca lo hizo innecesario. Cada mejora se

[3] Nota del traductor: En inglés, el vocablo *free* es polisémico, pudiéndose entender *free software* como software gratuito o como software libre. De ahí la aclaración que hace el autor.

añadía a la versión estándar que se distribuía, y acababa estando disponible para el público en general. Los clientes me pagaban para que realizase las mejoras que ellos querían, en lugar de las que yo habría considerado como más prioritarias.
La Free Software Foundation, una fundación exenta de impuestos para el desarrollo de software libre, obtiene sus ingresos mediante la venta de CD-ROM, camisetas y manuales (todos los cuales pueden ser copiados y alterados libremente por los usuarios), y mediante las donaciones que recibe. Actualmente tiene una plantilla de cinco programadores, más tres empleados que gestionan las peticiones por correo.
Algunos desarrolladores de software libre obtienen sus ingresos de la venta de servicios de soporte. Cygnus Solutions, con unos 50 empleados, estima que alrededor del 15 por ciento de la actividad de su plantilla se dedica a la realización y mejora de software libre—un porcentaje respetable para una compañía de software.
Varias compañías, incluyendo Intel, Motorola, Texas Instruments y Analog Devices, se han aliado para financiar el mantenimiento continuado del compilador libre de GNU para el lenguaje C. Mientras tanto, el ejército del aire de los EEUU está financiando el compilador de GNU para el lenguaje Ada, por pensar que ésta es la forma más económica de obtener un compilador de calidad. (La financiación terminó hace algún tiempo; el compilador de Ada de GNU está actualmente funcionando, y su mantenimiento se financia comercialmente).
Todos éstos son pequeños ejemplos. El movimiento del software libre es aún reducido y joven. Pero el ejemplo de las cadenas de radio mantenidas por los oyentes de este país (EEUU) muestra que es posible mantener una gran actividad sin forzar a que cada usuario pague.

Como usuario actual de computadoras, puede que estés utilizando un programa propietario. Si tu amigo te pidiese una copia, estaría mal que te negases a hacérsela. La cooperación es más importante que el *copyright*. Pero la cooperación clandestina, encubierta, no contribuye a formar una buena sociedad. Cualquier persona debería aspirar a vivir abiertamente, erguido, con orgullo, y esto significa decir *No* al software propietario.

Mereces poder cooperar abierta y libremente con otras personas que utilizan software. Mereces poder aprender cómo funciona el software, y enseñar a tus estudiantes con él. Mereces poder contratar a tu programador favorito para arreglarlo cuando falle.

Te mereces el software libre.

# El derecho a leer

Richard Stallman

Este artículo apareció en el número de febrero de 1.997 de Communications of the ACM (volumen 40, número 2)
Traducido del original en inglés por Pedro de las Heras Quirós y Jesús M. González Barahona

**(Tomado de “La ruta hacia Tycho”, una colección de artículos sobre los antecedentes de la Revolución Lunar, publicada en Luna City, en el año 2096)**

El camino hacia Tycho comenzó para Dan Halbert en la Facultad, cuando Lissa Lenz le pidió que le dejara su computadora. La suya se había averiado, y si no se la pedía a alguien no podría terminar el proyecto semestral. Ella no se habría atrevido a pedírsela a nadie, excepto a Dan.

Esto situó a Dan ante un dilema. Tenía que ayudarle pero, si le prestaba su computadora, ella podría leer sus libros. Además de poder ir a prisión durante muchos años por dejar que alguien leyese sus libros, la misma idea de hacerlo le escandalizó al principio. Igual que a todo el mundo, le habían enseñado desde el parvulario que compartir los libros era repugnante y equivocado, algo que sólo haría un pirata.

Y era muy probable que la SPA (Software Protection Authority, Autoridad para la Protección del Software) les cogiese. Dan había aprendido en su clase de Software que cada libro tenía un chivato de copyright que informaba a la Central de Licencias de quién, dónde y cuándo lo leía. (Esta información se utilizaba para coger a piratas de la lectura, pero también para vender perfiles de intereses personales a comerciantes). La próxima vez que su computadora se conectase a la red la Central de Licencias sería informada. Él, como dueño de una computadora, podría recibir el castigo más severo, por no tomar medidas para prevenir el delito.

Por supuesto, podría ser que Lissa no quisiera leer sus libros. Podría querer la computadora sólo para escribir su proyecto. Pero Dan sabía que ella era de una familia de clase media, y que a duras penas podía pagar la matrícula, y menos aún las cuotas de lectura. Puede que leer los libros de Dan fuese para ella la única forma de terminar los estudios. Sabía lo que era eso: él mismo había tenido que pedir un préstamo para poder pagar los artículos de investigación que leía. (El 10 % de los ingresos por ese concepto iba a parar a los investigadores que habían escrito los artículos. Como Dan pretendía dedicarse a la investigación, tenía esperanzas de que algún día sus propios artículos, si eran citados frecuentemente, le proporcionarían el dinero necesario para pagar el préstamo.)

Más tarde Dan supo que había habido un tiempo en el que cualquiera podía ir a una biblioteca y leer artículos de revistas especializadas, e incluso libros, sin tener que pagar. Había estudiantes independientes que leían miles de páginas sin tener becas de biblioteca del Gobierno. Pero en los años noventa, tanto los editores de revistas sin ánimo de lucro como los editores comerciales, habían

comenzado a cobrar cuotas por el acceso a sus publicaciones. Hacia el año 2047 las bibliotecas que ofrecían acceso libre a la literatura académica eran un recuerdo lejano.

Naturalmente había formas de engañar a la SPA y a la Central de Licencias. Eran, por supuesto, ilegales. Dan había tenido un compañero en la clase de Software, Frank Martucci, que había conseguido una herramienta ilegal de depuración, y la había utilizado para saltarse el código del chivato de copyright cuando leía libros. Pero se lo había contado a demasiados amigos, y uno de ellos le delató a la SPA para obtener una recompensa (los estudiantes muy endeudados eran fácilmente tentados por la traición). En 2047 Frank estaba en la cárcel, no por practicar la piratería de la lectura, sino por poseer un depurador.

Dan supo más tarde que hubo un tiempo en el que cualquiera podía poseer herramientas de depuración. Incluso había herramientas de depuración libres, disponibles en CD, o en la red. Pero los usuarios normales comenzaron a utilizarlas para saltarse los chivatos de copyright, y llegó un momento en que un juez estimó que éste se había convertido en el principal uso de los depuradores. Esto provocó que pasasen a ser ilegales, y se encarcelara a los que los desarrollaban.

Naturalmente, los programadores aún necesitaban herramientas de depuración, pero en el año 2047 los vendedores de depuradores sólo distribuían copias numeradas, y sólo a los programadores que tuvieran una licencia oficial, y que hubiesen depositado la fianza preceptiva para cubrir posibles responsabilidades penales. El depurador que utilizó Dan en la clase de software estaba detrás de un cortafuegos especial para que sólo lo pudiese utilizar en los ejercicios de clase.

También era posible saltarse los chivatos de copyright si se instalaba un kernel modificado. Más adelante, Dan supo que habían existido kernels libres, incluso sistemas operativos completos libres, hacia el fin del siglo anterior. Pero no sólo eran ilegales, como los depuradores, sino que no se podían instalar sin saber la contraseña del superusuario del sistema. Y ni el FBI ni el Servicio de Atención de Microsoft iban a decírtela.

Dan acabó por concluir que no podía dejarle la computadora a Lissa. Pero tampoco podía negarse a ayudarle, porque estaba enamorado de ella. Le encantaba hablar con ella. Y el que le hubiera escogido a él para pedir ayuda podía significar que ella también le quería.

Dan resolvió el dilema haciendo algo aún más inimaginable: le dejó la computadora, y le dijo su contraseña. De esta forma, si Lissa leía sus libros, la Central de Licencias creería que era él quién los estaba leyendo. Aunque era un delito, la SPA no podría detectarlo automáticamente. Sólo se darían cuenta si Lissa se lo decía.

Por supuesto, si la Facultad supiese alguna vez que Dan le había pasado a Lissa su propia contraseña, sería el final para ambos como estudiantes, independientemente de para qué la hubiese utilizado ella. La política de la Facultad era que cualquier interferencia con los medios que se usaban para realizar seguimientos del uso de las computadoras por parte de los estudiantes era motivo

suficiente para tomar medidas disciplinarias. No importaba si se había causado algún daño: la ofensa consistía en haber dificultado el seguimiento por parte de los administradores. Asumían que esto significaba que estabas haciendo alguna otra cosa prohibida y no necesitaban saber qué era.

Los estudiantes no solían ser expulsados por eso. Al menos no directamente. Se les prohibía el acceso al sistema de computadoras de la Facultad, por lo que inevitablemente suspendían todas las asignaturas.

Posteriormente Dan supo que este tipo de política universitaria comenzó en la década de los ochenta del siglo pasado, cuando los estudiantes universitarios empezaron a utilizar masivamente las computadoras. Anteriormente, las universidades mantenían una política disciplinaria diferente: castigaban las actividades que eran dañinas, no aquéllas que eran simplemente sospechosas.

Lissa no delató a Dan a la SPA. La decisión de Dan de ayudarle les condujo al matrimonio, y también a cuestionarse las enseñanzas que habían recibido de pequeños sobre la piratería. La pareja comenzó a leer sobre la historia del copyright, sobre la Unión Soviética y sus restricciones para copiar, e incluso la Constitución original de los Estados Unidos. Se trasladaron a Luna City, donde encontraron a otros que también se habían apartado del largo brazo de la SPA. Cuando la sublevación de Tycho comenzó en 2062, el derecho universal a la lectura se convirtió en uno de sus objetivos principales.

**Nota del autor:**

El derecho a la lectura es una batalla que se libra en nuestros días. Aunque pueden pasar 50 años hasta que nuestra forma de vida actual quede sumida en la oscuridad, muchas de las leyes y prácticas descritas en este relato han sido propuestas, ya sea por el gobierno de Clinton, en EEUU, o por las empresas editoriales.

Sólo hay una excepción: la idea de que el FBI y Microsoft tengan (y oculten) la contraseña de administración de las computadoras. Ésta es una extrapolación de las propuestas sobre el chip Clipper y otras propuestas similares de custodia de clave (key-escrow) del gobierno de Clinton, y de una tendencia que se mantiene desde hace tiempo: los sistemas informáticos se preparan, cada vez más, para proporcionar a operadores remotos el control sobre la gente que realmente utiliza los sistemas.

La SPA, que en realidad son las siglas de Software Publisher's Association (Asociación de Editores de Software), no es hoy día, oficialmente, una fuerza policial. Sin embargo, oficiosamente, actúa como tal. Invita a la gente a informar sobre sus compañeros y amigos. Al igual que el gobierno de Clinton, promueve una política de responsabilidad colectiva, en la que los dueños de computadoras deben hacer cumplir activamente las leyes de copyright, si no quieren ser castigados.

La SPA está amenazando a pequeños proveedores de Internet, exigiéndoles que permitan a la SPA espiar a todos los usuarios. Muchos proveedores se rinden cuando les amenazan, porque no puede permitirse litigar en los tribunales.

(Atlanta Journal-Constitution, 1 de octubre de 1996, D3.) Al menos un proveedor, Community ConneXion de Oakland, California, rechazó la exigencia y actualmente ha sido demandado Se dice que la SPA ha abandonado este pleito recientemente, aunque piensan continuar la campaña por otras vías.

Las políticas de seguridad descritas en el relato no son imaginarias. Por ejemplo, una computadora en una de las universidades de la zona de Chicago muestra en la pantalla el siguiente mensaje cuando se entra en el sistema (las comillas están en el original en inglés):

"Este sistema sólo puede utilizarse por usuarios autorizados. Las actividades de los individuos que utilicen este sistema informático sin autorización o para usos no autorizados pueden ser seguidas y registradas por el personal a cargo del sistema. Durante el seguimiento de individuos que estén usando el sistema inadecuadamente, o durante el mantenimiento del sistema, pueden ser seguidas también las actividades de usuarios autorizados. Cualquiera que use este sistema consiente expresamente ese seguimiento y es advertido de que si dicho seguimiento revela evidencias de actividad ilegal o violaciones de las ordenanzas de la universidad, el personal a cargo del sistema puede proporcionar las pruebas fruto de dicho seguimiento a las autoridades universitarias y/o a los agentes de la ley."

Ésta es una aproximación interesante a la Cuarta Enmienda de la Constitución de EEUU: presiona a todo el mundo, por adelantado, para que ceda en sus derechos.

## Bibliografía

The Copyright Grab, Pamela Samuelson, Wired, Jan. 1996
http://wwww.wired.com/wired/4.01/features/white.paper.html

Sold Out, James Boyle, New York Times, 31 March 1996
http://www.ese.ogi.edu/sold.out.html

Union for the Public Domain (Unión por el Dominio Público) es una organización nueva que pretende resistir e invertir la tendencia a la aplicación exagerada de la propiedad intelectual. Para más información, consultar:
http://www.public-domain.org.

## Biografía

Richard Stallman recibió el premio Grace Murray Hopper de la ACM en 1990, por el desarrollo de GNU Emacs. Es también autor del depurador simbólico libre GDB, y fundador del proyecto para el desarrollo del sistema operativo libre GNU.

# Definición de Software Libre

Free Software Foundation

Extraído de la página web de la Free Software Foundation **
Traducido por el equipo de traductores al español del proyecto GNU

Mantenemos esta definición de software libre para mostrar claramente qué debe cumplir un programa de software concreto para que se le considere software libre.

El software libre es un asunto de libertad, no de precio. Para entender el concepto, debes pensar en *libre* como en *libertad de expresión*, no como en *barra libre*[1].

Software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, deberías tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar[2]. El ser libre de hacer esto significa (entre otras cosas) que no tienes que pedir o pagar permisos.

También deberías tener la libertad de hacer modificaciones y utilizarlas de manera privada en tu trabajo u ocio, sin ni siquiera tener que anunciar que dichas modificaciones existen. Si publicas tus cambios, no tienes por qué avisar a nadie en particular, ni de ninguna manera en particular.

La libertad para usar un programa significa la libertad para cualquier persona u organización de usarlo en cualquier tipo de sistema informático, para cualquier clase de trabajo, y sin tener obligación de comunicárselo al desarrollador o a alguna otra entidad específica.

---

** `http://www.gnu.org/philosophy/free-sw.es.html`

[1] Nota del traductor: en inglés una misma palabra (free) significa tanto libre como gratis, lo que ha dado lugar a cierta confusión.

[2] `http://www.gnu.org/philosophy/free-sw.es.html\#exportcontrol`

La libertad de distribuir copias debe incluir tanto las formas binarias (código máquina) o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar (distribuir programas de modo ejecutable es necesario para que los sistemas operativos libres sean fáciles de instalar). Está bien si no hay manera de producir un binario o ejecutable de un programa concreto (ya que algunos lenguajes no tienen esta capacidad), pero debes tener la libertad de distribuir estos formatos si encontraras o desarrollaras la manera de crearlos.

Para que las libertades de hacer modificaciones y de publicar versiones mejoradas tengan sentido, debes tener acceso al código fuente del programa. Por lo tanto, la posibilidad de acceder al código fuente es una condición necesaria para el software libre.

Para que estas libertades sean reales, deben ser irrevocables mientras no hagas nada incorrecto; si el desarrollador del software tiene el poder de revocar la licencia aunque no le hayas dado motivos, el software no es libre.

Son aceptables, sin embargo, ciertos tipos de reglas sobre la manera de distribuir software libre, mientras no entren en conflicto con las libertades centrales. Por ejemplo, el copyleft es la regla que implica que cuando se redistribuya el programa no se pueden agregar restricciones para denegar a otras personas las libertades centrales. Esta regla no entra en conflicto con las libertades centrales, sino que más bien las protege.

Así pues, quizás hayas pagado para obtener copias de software GNU, o tal vez las hayas obtenido sin ningún coste. Pero independientemente de cómo hayas conseguido tus copias, siempre tienes la libertad de copiar y modificar el software, e incluso de vender copias.

Software libre no significa *no comercial*. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial. El desarrollo comercial del software libre ha dejado de ser inusual; el software comercial libre es muy importante.

Es aceptable que haya reglas acerca de cómo empaquetar una versión modificada, siempre que no bloqueen a consecuencia de ello tu libertad de publicar versiones modificadas. Reglas como "Si haces disponible el programa de esta manera, debes hacerlo disponible también de esta otra" pueden ser igualmente aceptables, bajo la misma condición (observa que una regla así todavía te deja decidir si publicar o no el programa). También es aceptable que la licencia requiera que, si has distribuido una versión modificada y el desarrollador anterior te pide una copia de ella, debas enviársela.

En el proyecto GNU, utilizamos el copyleft para proteger de modo legal estas libertades para todos. Pero el software libre sin copyleft también existe. Creemos que hay razones importantes por las que es mejor usar el copyleft, pero si tus programas son software libre sin ser copyleft, los podemos utilizar de todos modos.

Visita la página con las categorías de software libre[3] para ver una descripción de las diferencias que hay entre el software libre, software copyleft y otras categorías de software y cómo se relacionan unas con otras.

A veces las normas de control de exportación del gobierno y las sanciones mercantiles pueden restringir tu libertad de distribuir copias de los programas a nivel internacional. Los desarrolladores de software no tienen el poder de eliminar o sobrepasar estas restricciones, pero lo que pueden y deben hacer es rehusar en imponerlas como condiciones de uso del programa. De esta manera, las restricciones no afectarán a actividades y gente fuera de las jurisdicciones de estos gobiernos.

Cuando se habla de software libre, es mejor evitar términos como *regalar* o *gratis*, porque esos términos implican que lo importante es el precio, y no la libertad. Algunos términos comunes tales como *piratería* conllevan opiniones que esperamos que no apoyes. Visita la página de palabras y frases confusas[4] que vale la pena evitar, donde encontrarás una discusión acerca de estos términos. También tenemos una lista de traducciones de software libre[5] a varios idiomas.

Por último, fíjate en que los criterios establecidos en esta definición de software libre requieren pensarse cuidadosamente para ser interpretados. Para decidir si una licencia de software concreta es una licencia de software libre, lo juzgamos basándonos en estos criterios para determinar si tanto su espíritu como su letra en particular los cumplen. Si una licencia incluye restricciones contrarias a nuestra ética, la rechazamos, aun cuando no hubiéramos previsto el problema en estos criterios. A veces un requisito de una licencia plantea una situación que necesita de una reflexión minuciosa, e incluso conversaciones con un abogado, antes de que podamos decidir si la exigencia es aceptable. Cuando llegamos a una conclusión, a veces actualizamos estos criterios para que sea más fácil ver por qué ciertas licencias se pueden calificar o no como de software libre.

Si te interesa saber si una licencia concreta es de software libre, mira nuestra lista de licencias[6]. Si la licencia que te preocupa no está en la lista, puedes preguntarnos enviándonos un correo electrónico a licensing@gnu.org.



---

[3] `http://www.gnu.org/philosophy/categories.es.html`

[4] `http://www.gnu.org/philosophy/words-to-avoid.es.html`

[5] `http://www.gnu.org/philosophy/fs-translations.html`

[6] `http://www.gnu.org/licenses/license-list.html`



# Apéndice

# Glosario de términos y acrónimos

**AbiWord**: Procesador de textos libre. Está integrado en GNOME, aunque también existen versiones para otras plataformas, incluido Windows. Su página web se puede encontrar en http://www.abisource.com.

**Ada**: Lenguaje de programación diseñado con la seguridad en mente, principal razón por la cual fue encargado por el Departamento de Defensa de EEUU. Es por eso que tiene una gran aceptación en la industria aeronáutica y aeroespacial. Su nombre proviene de Ada Lovelace, la primera hacker de la historia. El compilador más popular de Ada es GNAT.

**Apache**: Servidor web más popular con una cuota de mercado superior al 60 % desde hace años. Véase también Apache Software Foundation.

**Apache Software Foundation**: Fundación que se encarga de velar por el desarrollo y la promoción del servidor web Apache y de otros proyectos generalmente relacionados con tecnologías web como Jakarta. Más información en http://www.apache.org.

**Arquitectura** (de ordenadores): Se trata de un concepto que engloba el diseño y funcionamiento de los ordenadores que especifican entre otros aspectos el formato y el conjunto de instrucciones. Existen muchas arquitecturas, siendo la más popular la i386 por ser la utilizada en los PC.

**Arranque dual**: Al arrancar el ordenador, el usuario puede elegir entre varios sistemas operativos, como por ejemplo entre Windows y GNU/Linux. Cada sistema operativo ha de estar en una partición independiente.

**BarraPunto**: Sitio de noticias en español centrado en la temática de software libre, aunque no de manera exclusiva. Se puede visitar en http://www.barrapunto.com.

**BerliOS**: Portal tipo SourceForge. Puede encontrar más información sobre BerliOS en http://developer.berlios.de.

**bash**: Tipo de shell, en particular la que suele venir por defecto en la mayoría de los sistemas GNU/Linux.

**BSA**: Acrónimo de Business Software Alliance. Se trata de una asociación de la que son miembros la mayoría de las grandes compañías de software propietario que lucha contra la copia ilegal. Su lema es *Promoviendo un mundo digital seguro y legal*.

**BSD**: Acrónimo de Berkeley Software Distribution (Distribución de Software de Berkeley). Da nombre tanto a sistemas como a un tipo de licencias. Los sistemas BSD son sistemas libres basados en Unix, pero con un núcleo y un conjunto de herramientas ligeramente diferentes a las que encontramos en GNU/Linux. Existen varios *sabores* de BSD: FreeBSD, OpenBSD y NetBSD, cada uno con sus peculiaridades. Las licencias BSD son también conocidas como licencias minimalistas.

***BSD**: véase BSD.

**C/C++**: Lenguajes de programación. C fue creado a principios de los 70 para el desarrollo de Unix; se trata de uno de los primeros lenguajes de programación que permiten la programación de alto nivel. C++ es una versión posterior de C que añade técnicas modernas de programación. La mayoría del código libre está escrito en C o C++.

**ChatZilla**: Cliente de IRC. IRC es un sistema que permite la comunicación síncrona a través de Internet. ChatZilla se desarrolla como parte del proyecto Mozilla. Más información sobre el proyecto ChatZilla en http://www.mozilla.org/projects/rt-messaging/chatzilla.

**COBOL**: Acrónimo de Common Organization Business Oriented Language (Lenguaje Común Orientado a la Organización de Negocios), aunque también se puede encontrar como COmmon Business Oriented Language (Lenguaje Común Orientado a Negocios). Se trata de un lenguaje de programación de finales la década de los 60 cuyo propósito era poder ser utilizado en cualquier ordenador. A día de hoy su utilización es bastante limitada, aunque se encuentra con frecuencia en aplicaciones bancarias y programas antiguos.

**Código abierto**: Véase Open Source.

**Código fuente** (también conocido como fuentes): Se trata de las instrucciones de ordenador escritas en un lenguaje de programación. En la fase de compilación se transforma en código máquina. Para que el software sea libre, el código fuente debe ser accesible, ya que sino la posibilidad de realizar modificaciones, aunque no sea imposible, se dificulta sobremanera.

**Código máquina** (también conocido como código binario): Se trata del código que los ordenadores pueden ejecutar. Consta de unos y ceros, aunque existen otras formas de representación como octal o hexadecimal. El código máquina es difícilmente comprensible para los humanos -y la creación de complejas aplicaciones casi imposible-, por lo que se crearon los lenguajes de programación de alto nivel.

**Compilador**: Se encarga principalmente de traducir los ficheros escritos en lenguajes de programación (comprensibles para los humanos) en código máquina (unos y ceros, comprensibles generalmente sólo por los ordenadores). Ejemplos de compiladores son GCC y GNAT.

**Compresión**: Las secuencias de audio y vídeo ocupan mucho espacio, por lo que se utilizan técnicas de compresión que hacen que el tamaño disminuya. De esta forma se facilita su intercambio (el tiempo de descarga de un vídeo comprimido en formato DivX es menor que sin comprimir o un vídeo que antes requería mucho espacio ahora cabe en un CD-ROM). DivX y MP3 comprimen con pérdidas, lo que significa que el archivo comprimido no tiene la calidad del original. Sin embargo, las pérdidas son -en muchas ocasiones y dependiendo del factor de pérdidas introducido- tolerables (incluso indetectables) para el ser humano. Existen formatos de compresión sin pérdidas, como el usado en los archivos zip, pero su utilidad es menor para audio y vídeo.

**Copyleft**: Tipo de licencia que obliga a los que redistribuyen el software a hacerlo bajo las mismas condiciones con las que lo recibieron. De esta forma, se transfiere a quien recibe cualquier trabajo derivado las mismas libertades de redistribución y modificación que le dieron al original. El lema oficioso del copyleft es *all rights reversed*. La licencia más conocida es la GNU GPL, aunque existe alguna más. Más información en http://www.gnu.org/copyleft/copyleft.es.html.

**CORBA**: Acrónimo de Common Object Request Broker Architecture. Es un estándar que permite la intercomunicación entre aplicaciones heterogéneas. De esta forma, programas escritos en diferentes lenguajes de programación y para arquitecturas diferentes pueden interoperar. Más información en http://www.corba.org.

**Cracker**: Persona (habilidosa o no con entornos informáticos) con intenciones *maliciosas*. Sinónimo de *delincuente informático*. Nótese la diferencia con hacker.

**CVS**: Acrónimo de Concurrent Versions System (Sistema Concurrente de Versiones). Sistema que permite a varios desarrolladores trabajar de manera simultánea y asíncrona en la programación de un mismo software facilitando la sincronización y minimizando las colisiones. Más información en http://www.cvshome.org.

**DB2**: Gestor de base de datos creado por IBM.

**DeCSS**: CSS es el acrónimo para Content Scrambling System (Sistema de Cifrado de Contenidos), un sistema de cifrado utilizado en DVDs. El principal propósito de CSS es evitar la copia ilegal, pero también puede forzar otras restricciones. DeCSS es un programa que rompe este cifrado y permite la lectura de estos DVDs. El noruego Jon Johansen fue llevado a juicio por grandes corporaciones multimedia a finales de la década de los 90 por haber realizado una implementación de DeCSS para GNU/Linux. Fue absuelto a principio de 2004.

**Debian**: Sistema operativo libre gestionado y promocionado íntegramente por cerca de mil voluntarios. En la actualidad, Debian utiliza el kernel de Linux para llevar a cabo su distribución (aunque se espera que existan distribuciones Debian con otros kernels, por ejemplo con HURD, en el futuro). Actualmente está disponible para varias arquitecturas, entre ellas la i386. Más información en http://www.debian.org/index.es.html.

**Depurador**: Programa de ordenador utilizado a la hora de crear software para encontrar y corregir errores de programación. El depurador libre más conocido es GDB del proyecto GNU.

**Distribución**: La labor de las distribuciones es la integración de software independiente para su correcto funcionamiento en conjunto. Las distribuciones de GNU/Linux, como Debian, Red Hat, Slackware o SuSE, toman el código de los autores originales y lo *empaquetan* de manera que al usuario final le sea fácil instalar, actualizar, borrar y usar el software en su ordenador.

**Distro**: véase distribución.

**DivX**: Tecnología de compresión de vídeo basada en el estándar de compresión MPEG-4.

**DMCA**: Acrónimo de Digital Millenium Copyright Act. Se trata de una actualización de las leyes de copyright promulgadas por el Senado de EEUU a

finales de la década de los 90. El DMCA ha causado gran controversia, sobre todo por una sección escrita de manera tan amplia que podría dar pie a situaciones que limiten de manera efectiva la libertad de expresión.

**Drivers**: Software que se encarga de interactuar entre el sistema operativo y los dispositivos (hardware).

**DOS**: Acrónimo de Disk Operating System (Sistema Operativo de Disco). Sistema operativo de Microsoft lanzado a mediados de los años 80. Hoy en franco desuso.

**(GNU) Emacs**: Editor de textos (para algunos, algo más) creado por Richard Stallman como parte del proyecto GNU. Véase también XEmacs.

**Eazel**: Compañía dedicada al desarrollo de GNOME, en especial del gestor de archivos Nautilus. Creada en 1999, tuvo que cerrar a mediados de 2001 por quiebra.

**Eric Raymond**: Conocido hacker norteamericano conocido no sólo por el desarrollo de programas de software libre, sino por la autoría de varios artículos que plasman la filosofía más pragmática del software libre, en especial *La Catedral y el Bazar*. Fue uno de los propulsores de la Open Source Initiative y del término Open Source.

**Evolution**: Gestor de información personal para GNOME desarrollado principalmente por Ximian. Se trata de una aplicación que reune un potente cliente de correo electrónico, una agenda, un libro de contactos y un gestor de tareas. Más información en su página web: http://ximian.com/products/evolution.

**FreshMeat**: Sitio web muy concurrido donde se anuncian las nuevas versiones de aplicaciones y sistemas de software libre. Es propiedad de VA Linux. Su URL es http://www.freshmeat.net.

**FreeBSD**: Sistema tipo BSD. En la siguiente dirección hay más información en español http://www.freebsd.org/es/.

**Free Software Foundation** (Fundación del Software Libre): Entidad sin ánimo de lucro fundada a mediados de los 80 por Richard Stallman para promocionar el uso del software libre. Su proyecto más conocido es el proyecto GNU, que no sólo ha aportado gran cantidad de software sino que también se ha preocupado de difundir la filosofía del software libre. Más información en su página web en http://www.fsf.org.

**FSF**: Véase Free Software Foundation.

**Fundación GNOME**: Fundación creada en el 2000 para velar por los intereses y el desarrollo del proyecto GNOME. La Fundación GNOME tiene un consejo directivo formado por cerca de una docena de miembros que son elegidos democráticamente entre todos los participantes del proyecto GNOME, por lo que se trata de una organización basada en la meritocracia.

**Galeon**: Navegador web que utiliza el motor de Mozilla, Gecko. Es muy popular por su velocidad, ya que a diferencia de las suites de Internet que engloban todo tipo de aplicaciones (navegador, cliente de correo, generador de páginas web,

etc.), está especializado en la navegación. Su página web se puede encontrar en http://galeon.sourceforge.net.

**Gentoo**: Distribución de GNU/Linux un tanto especial, ya que sus paquetes software no vienen precompilados como es usual en el resto de distribuciones. Eso permite que el usuario de Gentoo pueda optimizarse y personalizarse de manera automática, lo que la ha hecho muy querida entre aquéllos que buscan potencia y configurabilidad. Más información en http://www.gentoo.org.

**Gecko**: motor de Mozilla, también utilizado por otros navegadores web (como Galeon). El motor de un navegador web se encarga de interpretar el código de las páginas web (escrito en HTML) y mostrar su contenido en la ventana del navegador.

**GCC**: Acrónimo de GNU Compiler Collection (Colección de Compiladores de GNU). Se trata de una serie de compiladores para diversos lenguajes de programación (C, C++, Java, etc.). Más información en http://gcc.gnu.org.

**GDB**: Acrónimo de GNU DeBugger (Depurador de GNU). Depurador del proyecto GNU desarrollado inicialmente por Richard Stallman a mediados de los años 80. GDB se puede utilizar para muchos lenguajes de programación, entre los que se encuentran C y C++.

**(The) GIMP**: Acrónimo de GNU Image Manipulation Program (Programa de Manipulación de Imágenes de GNU). Programa de tratamiento de imágenes similar a Adobe Photoshop. También existe una versión para Windows. Más información en http://www.gimp.org.

**GNAT**: Acrónimo de GNU Ada Translator (Traductor Ada de GNU). Compilador para el lenguaje Ada. Más información en http://www.gnat.com.

**GNOME**: Acrónimo de GNU Network Object Modelling Environment. Entorno de escritorio orientado a componentes CORBA cuyo objetivo es ofrecer al usuario final un interfaz amigable para GNU/Linux. Véase también KDE. Su página principal es http://www.gnome.org.

**(Proyecto) GNU**: Acrónimo recursivo de GNU's Not Unix. Proyecto lanzado por la Free Software Foundation con el objetivo de conseguir un sistema operativo similar a Unix, pero totalmente libre. Entre sus grandes logros está la articulación de la licencia GNU GPL. Más información en http://www.gnu.org.

**GNU/Linux**: Unión del kernel Linux y las herramientas proporcionadas por el proyecto GNU. Se trata de una solución de compromiso adoptada por la comunidad de software libre debido a que el fulgurante éxito de Linux ha propiciado que todo el sistema se llame como una de sus partes: el kernel.

**GNU GPL**: Acrónimo de GNU General Public License (Licencia Pública General de GNU). Se trata de la licencia copyleft más popular creada por la Free Software Foundation dentro del proyecto GNU. Se puede encontrar una traducción de la Licencia Pública General de GNU a nuestro idioma en http://www.garaitia.com/new/gpl-spanish.php.

**Gnumeric**: Hoja de cálculo para GNOME. Su autor original es Miguel de Icaza, aunque posteriormente su desarrollo fue llevado principalmente por Ximian. Más información en la página web de la aplicación: http://www.gnome.org/projects/gnumeric.

**GPL**: Véase GNU GPL.

**Hacker**: Programador habilidoso, experto en sistemas informáticos, gurú. Nótese la diferencia con cracker.

**Hardware**: Conjunto de dispositivos físicos que componen el ordenador: la pantalla, el teclado, el ratón, etc.

**HelixCode**: Pasó a llamarse Ximian Inc. en 2001 por un problema de nombres registrados.

**HispaLinux**: Asociación española de usuarios de software libre. Uno de sus grupos de interés más conocidos es ProInnova. Más información en http://www.hispalinux.es.

**HTML**: Acrónimo de HyperText Markup Language (Lenguaje de Marcado de HiperTexto). Es el lenguaje en el que están escritas las páginas web. Se trata de un subconjunto de SGML.

**(The) (GNU) HURD**: Kernel del proyecto GNU cuya pretensión es sustituir algún día a Linux. Actualmente en desarrollo. Más información: http://www.gnu.org/software/hurd/hurd.

**i386**: Arquitectura de ordenador típica de los ordenadores personales (PC).

**Jakarta**: Subproyecto del proyecto Apache cuyo objetivo es crear soluciones libres en Java, principalmente para el entorno web. Jakarta toma el nombre de la capital de la isla de Java, ya que el lenguaje de programación principal en el que está implementado es precisamente Java. Más información en http://jakarta.apache.org.

**Java**: Moderna plataforma de programación creada por SUN en la década de los años 90 que incluye un lenguaje de programación propio.

**KDE**: Acrónimo de K Desktop Environment (Entorno de Escritorio K). Entorno de escritorio completo cuya finalidad es acercar al usuario final a los sistemas GNU/Linux gracias a su amigabilidad y facilidad de manejo. Véase también GNOME. La página principal del proyecto KDE se puede encontrar en http://www.kde.org.

**KDevelop**: Entorno de desarrollo integrado para el proyecto KDE. Un entorno de desarrollo integrado está pensado para facilitar a los desarrolladores la tarea de creación de software. Generalmente incluye, entre otros elementos, un editor de texto, un compilador y un depurador.

**Kernel**: Núcleo del sistema operativo. Es el que se encarga de las labores de más bajo nivel (el nivel más cercano al hardware) tales como gestión de memoria, de entrada/salida de dispositivos, etc. El kernel más popular en el mundo del software libre es Linux, aunque hay muchos más (por ejemplo los sistemas BSD tienen uno propio).

**Knoppix**: Se trata de una distribución *live* de GNU/Linux basada en Debian. Las distribuciones *live* permiten hacer uso de un sistema sin necesidad de tenerlo instalado en el ordenador, ya que arrancan desde el CD y todas las aplicaciones utilizadas están incluidas en el propio CD. Este tipo de distribuciones se han

hecho muy populares en los últimos tiempos, ya que permiten probar software de manera sencilla. Más información en http://www.knoppix.de.

**KOffice**: Completa suite ofimática de KDE. Incluye, entre otros programas, un procesador de textos (KWriter), una hoja de cálculo (KSpread), un programa para realizar presentaciones (KPresenter) y una aplicación de flujo de grafos del tipo Microsoft Visio (Kivio). Más información en http://www.koffice.org

**Konqueror**: Navegador web y gestor de ficheros (entre otras muchas funcionalidades) del proyecto KDE. Es muy potente y versátil. En su página web se puede encontrar más información sobre esta aplicación: http://www.konqueror.org

**Kriptópolis**: Sitio web en español dedicado a la privacidad y seguridad en Internet. La dirección de su web es http://www.kriptopolis.com.

**Lenguaje de programación**: Conjunto de reglas semánticas y sintácticas utilizadas para dar instrucciones a un ordenador. Los lenguajes de programación permiten trabajar a un nivel de abstracción superior que con código máquina, lo que facilita la creación y mantenimiento de programas informáticos. Existen cientos, sino miles, de lenguajes de programación. Algunos ejemplos son C, C++, Ada, Java, Pascal y COBOL.

**LinEx**: Distribución patrocinada por la Junta de Extremadura para la difusión de las nuevas tecnologías en su territorio. Está basada en Debian. En Andalucía, la han tomado como base para realizar a su vez una distribución adaptada a sus necesidades y que se ha denominado GuadaLinEx. Más información en http://www.linex.org

**Linus Torvalds**: autor principal del kernel Linux. Linus comenzó a trabajar como entretenimiento en el desarrollo de un kernel de tipo Unix cuando era estudiante de informática en una universidad finlandesa a principios de la década de los 90. Con la popularización de Linux se trasladó al famoso Silicon Valley californiano donde ha estado trabajando primero para Transmeta, una compañía dedicada a la elaboración de chips, y posteriormente en Open Source Development Labs dedicado íntegramente al desarrollo de Linux.

**Linux**: Kernel de sistema operativo. Su autor principal es Linus Torvalds, aunque en su elaboración han ayudado miles de desarrolladores. Más información en http://www.kernel.org

**LuCAS**: Acrónimo de Linux en CAStellano. Se trata de un proyecto encaminado a ofrecer documentación sobre Linux en particular y sobre software libre en general en castellano. Su URL es http://lucas.hispalinux.es

**LSSI** (en realidad debería ser LSSI-CE): Acrónimo de Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico.

**Lynx**: Navegador web de texto. Se lanza desde la shell y permite navegar sin tener instalado un entorno gráfico. La página de Lynx (http://lynx.browser.org/), por supuesto, sólo contiene texto.

**(Licencia) Minimalista**: Tipo de licencia de software libre. Sin embargo, al contrario que las licencias copyleft (también conocidas como robustas), el programa puede ser redistribuido bajo las condiciones que se quiera siempre que

se mantenga la nota de autoría. Esto significa que alguien puede redistribuir un programa licenciado con una licencia minimalista como software propietario, si así lo desea. A las licencias minimalistas, también se las conoce como licencias BSD, ya que son estos sistemas los que las han hecho tan populares.

**Miguel de Icaza**: Hacker mexicano fundador del proyecto GNOME y de la compañía Ximian. Miguel de Icaza es tenido por uno de los grandes gurús del software libre.

**Mozilla**: Proyecto iniciado por la compañía Netscape a finales de la década de los 90 tras liberar su navegador Netscape Navigator. Mozilla es a día de hoy una suite de Internet que agrupa navegador, cliente de correo electrónico y compositor de páginas web. El proyecto Mozilla además proporciona un motor para páginas web Gecko y otra serie de herramientas muy populares, como por ejemplo ChatZilla. Más información en http://www.mozilla.org.

**MP3**: Formato de compresión para audio.

**NASDAQ**: Acrónimo de National Association of Securities Dealers Automated Quotation (difícilmente traducible). Se trata del mayor mercado bursátil de EEUU y el primer mercado de valores electrónico del mundo (se fundó en 1971). Es famoso por su apuesta por las nuevas tecnologías y muchas de las empresas más importantes del sector tecnológico e informático. Se puede seguir la evolución del NASDAQ desde http://www.nasdaq.com.

**Nautilus**: Gestor de ficheros del proyecto GNOME. Fue desarrollado originariamente por Eazel hasta que quebró. Gracias a la disponibilidad de su código fuente, un grupo ajeno a Eazel ha seguido con su desarrollo hasta la actualidad.

**Netscape**: es la compañía que a mediados de la década de los 90 comercializaba el popular navegador Netscape Navigator. Cuando ya daba por perdida la *guerra de los navegadores* con el Internet Explorer de Microsoft, publicó el código fuente de su navegador bajo una licencia de software libre y fundó el proyecto Mozilla con el objetivo de remontar el vuelo. Fue adquirida posteriormente por AOL. Hoy el proyecto Mozilla es totalmente independiente de Netscape.

**OCW**: Acrónimo de OpenCourseWare. Se trata de un proyecto del Instituto Tecnológico de Massachusetts orientado a ofrecer en Internet sus cursos docentes en un formato homogéneo. Su página web es http://web.mit.edu/ocw.

**Ofimática**: Encargada de la organización automatizada de información destinada a la administración de entornos de oficina. Generalmente consta de un procesador de textos, una hoja de cálculo y un sistema de bases de datos. Últimamente se le han añadido muchos otros elementos como programas para realizar presentaciones y demás. Ejemplos de herramientas ofimáticas libres son OpenOffice.org y KOffice.

**OpenOffice.org**: Suite ofimática libre desarrollada principalmente por SUN. Consta de un procesador de textos (Writer), una hoja de cálculo (Calc), un programa para presentaciones (Impress) y una aplicación para imágenes (Draw). Se puede encontrar más información sobre OpenOffice.org en http://www.openoffice.org.

**Open Source** (en español: código abierto): Denominación alternativa del software libre enfocada más en los aspectos pragmáticos (modelo de desarrollo más dinámico, productivo, de mejor calidad, etc.). Uno de los creadores de este término y de la Open Source Initiative que lo avala fue Eric Raymond. Véase http://www.opensource.org.

**OS/2**: es un sistema operativo desarrollado por IBM, en un principio pensado como sucesor de DOS. Tiene algunas características comunes tanto con Windows como con Unix.

**OSI**: véase Open Source Initiative.

**Open Source Initiative**: Organización sin ánimo de lucro dedicada a gestionar y promocionar el término Open Source. Su página web es http://www.opensource.org.

**PADRE**: Acrónimo de Programa de Ayuda para efectuar la Declaración de la Renta en España.

**Partición**: División del espacio de almacenamiento del disco duro en partes independientes. Esto posibilita, entre otras funcionalidades, tener sistemas operativos diferentes en un mismo ordenador. Véase también arranque dual.

**Pascal**: Lenguaje de programación de la década de los 70 escasamente utilizado hoy en día, aunque algunos de sus *sucesores* sí que cuentan con amplio eco en la industria del software.

**plug-and-play**: Método cuyo objetivo es que el sistema operativo, los controladores y los elementos hardware funcionen de manera conjunta sin necesidad de la intervención del usuario.

**ProInnova**: Grupo de interés de HispaLinux y de la Asociación de Técnicos de Informática (ATI) en favor de la libertad de innovación. Sus esfuerzos, por el momento, están centrados en dos problemas: patentes de software y extensiones de las legislaciones sobre derechos de autor. Más información en http://proinnova.hispalinux.es.

**Red Hat**: Distribución de GNU/Linux. Se trata de una distribución comercial -gestionada por la compañía Red Hat Linux- que utiliza paquetes en formato RPM.

**Red Hat Linux**: Empresa que realiza y comercializa Red Hat. Fue una de las primeras empresas de software libre en aparecer en medios de comunicación no dedicados específicamente a la tecnología, en espacial tras su impresionante salida a bolsa en el índice NASDAQ a finales de la década de los 90. Aunque sus acciones a día de hoy valen una centésima parte del valor de pico que alcanzó antes del desastre de las *punto com,* desde hace un par de años tiene un balance contable positivo. La dirección de su página web es http://www.redhat.com.

**RIAA**: Acrónimo de Recording Industry Association of America (Asociación Americana de la Industria Discográfica). Entidad equivalente a la SGAE española. Su página web se puede encontrar en la siguiente dirección: http://www.riaa.org.

**Richard Stallman**: Gurú, hacker y filósofo del movimiento del software libre. A mediados de los años 80 fundó la Free Software Foundation y el proyecto GNU. Ha participado en la creación de multitud de programas de software libre (el editor de textos Emacs, el depurador GDB, etc.). En los últimos años, su actividad

principal se limita a viajar por el mundo y dar conferencias sobre software libre y en contra de la patentabilidad del software.

**(Licencia) Robusta**: Véase copyleft.

**RPM**: Acrónimo de Red Hat Package Manager (Gestor de Paquetes de Red Hat). Sistema de paquetes creado y utilizado en Red Hat y distribuciones derivadas. Una aplicación software suele empaquetarse en uno o varios paquetes para facilitar su instalación y configuración.

**Savannah**: Sitio tipo SourceForge del proyecto GNU. Puede encontrarse en http://savannah.gnu.org.

**SGAE**: Acrónimo de Sociedad General de Autores y Editores.

**SGML**: Acrónimo de Standard Generalized Markup Language. SGML es un metalenguaje que permite la creación de lenguajes de marcado, como pueden ser HTML o XML.

**Shell**: También conocido como la línea de instrucciones (o de comandos). Mientras en los entornos de ventanas, el sistema software espera la introducción de instrucciones por parte del usuario principalmente mediante el uso del puntero de ratón u otros elementos gráficos, en la línea de instrucciones las instrucciones son órdenes escritas mediante el teclado de manera textual.

**Sistema de ficheros**: Se encargan de gestionar el almacenamiento, organización jerárquica, manipulación, navegación, acceso y consulta de ficheros.

**Slackware**: Popular distribución de GNU/Linux, en muchas ocasiones recomendada a los novatos por su simplicidad de instalación y la inclusión de las últimas versiones de software libre publicado. Más información en http://www.slackware.com.

**Software**: Componente intangible en la informática. Generalmente se trata de una serie de instrucciones elaboradas por humanos en lenguajes de programación de alto nivel (código fuente) que luego son traducidas por un compilador a código máquina (unos y ceros comprendidos por las máquinas). El software se divide en software de sistema, parte que corresponde a los sistemas operativos, o de aplicación, que agrupa a los programas de los que el usuario suele hacer uso. Estrictamente el software también incluye la documentación del programa, aunque ésta se encuentre en un manual.

**Software libre**: Tipo de software con condiciones de uso y distribución que cumplen con las propiedades para ser considerado libre. En el artículo *Definición de Software Libre* incluido en esta colección se pueden encontrar las cuatro libertades que ha de tener un software para ser considerado libre.

**Software propietario** (o software privativo): Software con condiciones de uso y distribución que no cumplen con las condiciones para ser software libre. Véase software libre.

**StarOffice**: suite ofimática *hermana* de OpenOffice.org. OpenOffice.org es una bifurcación de una versión de StarOffice de principios de los 2000 que se publicó bajo las condiciones de software libre. SUN es la que se encarga del mantenimiento y de liderar el desarrollo tanto de StarOffice como de OpenOffice.org, aunque en el segundo caso -gracias a su licencia de software libre- se beneficie del

trabajo de la comunidad que participa en el desarrollo y promoción de OpenOffice.org.

**SuSE**: Acrónimo alemán de System und Software-Entwicklung (Desarrollo de Software y Sistemas). Distribución que tuvo sus orígenes a mediados de los 90 en Alemania. A finales de 2003, SuSE fue comprada por la norteamericana Novel. Su página web es http://www.suse.de

**Streaming** (en español: flujo): Se trata de un modo de transmisión de datos entre cliente y servidor un poco peculiar (para lo que es común en Internet) ideado principalmente para la transmisión de datos multimedia. El flujo de datos ha de ser constante, para evitar interrupciones incómodas en audio y vídeo, y se ha de mantener la ordenación original de los datos, porque sino se dificulta la comprensión de audio y vídeo. El streaming suele utilizarse conjuntamente con técnicas de compresión.

**SourceForge**: Sitio web que facilita la creación de software libre por parte de desarrolladores distribuidos por todo el mundo. Facilita espacio web para anunciar el proyecto, listas de correo electrónico, CVS, sistemas de gestión de erratas, etc. SourceForge es gestionado por la compañía VA Linux. En los últimos años han aparecido otros sitos tipo SourceForge, como Savannah y BerliOS. Se puede visitar SourceForge en http://www.sourceforge.net.

**tcsh**: Tipo de shell.

**Tiempo de ejecución**: Se trata del espacio temporal en el que el ordenador está ejecutando las instrucciones correspondientes a un programa software.

**Unix**: Sistema operativo muy portable (se puede usar en varias arquitecturas) creado a principios de los 70 en los laboratorios de AT&T por Ken Thompson, Dennis Ritchie y Dougles McIlroy. La filosofía de Unix han dado pie a una amplia gama de sistemas operativos que siguen sus principios, como los sistemas GNU/Linux, OS/2, etc. El lenguaje de programación C fue creado para el desarrollo de Unix.

**UCITA**: Acrónimo de Uniform Computer Information Transactions Act. Es una ley propuesta en EEUU para regular las transacciones de información entre ordenadores. Su complejidad hace que incluso sea difícil de comprender para los abogados. Una de las críticas más comunes a la UCITA es que favorece a las grandes compañías de software propietario, como las asociadas a la BSA.

**USB**: Acrónimo de Universal Serial Bus (Bus Serie Universal). Proporciona un bus serie estándar para conectar dispositivos al ordenador, de manera que la conexión de éstos se facilita enormemente. Más información en http://www.usb.org.

**VA Linux**: Empresa estadounidense que gestiona FreshMeat y SourceForge entre otros sitios web importantes en el mundo del software libre.

**WINE**: Acrónimo de WINE Is Not an Emulator. Permite ejecutar programas para Windows en un entorno Unix como pueden ser los sistemas GNU/Linux o los *BSD. Se puede encontrar más información en http://www.winehq.com.

**X**: Estándar *de facto* para sistemas de ventanas multiplataforma. La Fundación X.org es la que gestiona este estándar, además de promocionar el sistema de

ventanas X Window. Por eso, a las X Window se las conoce también popularmente como *las X*. Más información en http://www.x.org.

**XEmacs**: editor de texto basado en GNU Emacs. Se trata de una bifurcación de GNU Emacs, ya que algunos desarrolladores consideraron que la forma de trabajar de este proyecto no era la adecuada y decidieron crear otro. Su página principal es http://www.xemacs.org.

**Ximian**: Empresa fundada por Miguel de Icaza para acelerar el desarrollo del entorno de escritorio GNOME. En agosto de 2003 fue adquirida por Novell. Su página web se puede encontrar en http://www.ximian.com

**XFree86**: Implementación libre del sistema de ventanas X Window para GNU/Linux, los sistemas *BSD y otros. Más información en http://www.xfree86.org.

**XML**: Acrónimo de eXtensbile Markup Language (Lenguaje de Marcado eXtensible). Se trata de un subconjunto de SGML que permite describir datos. Ha ganado en popularidad en los últimos tiempos debido a que facilita el intercambio y procesamiento de datos. Más información en http://www.xml.com.

**X Window**: Sistema de ventanas para GNU/Linux y otros *sabores* de Unix. X Window fue lanzado a principios de los años 80 por el Massachusetts Institute of Technology. Los desarrolladores de las X Window hacen especial hincapié en que el nombre de su sistema de ventanas no incluye una ese al final. También se las conoce popularmente como *las X*. Véase también X y XFree86.

# Índice de Autores